1991 / NOVEMBER

ÁRA: 196 FT

ALAPLAP

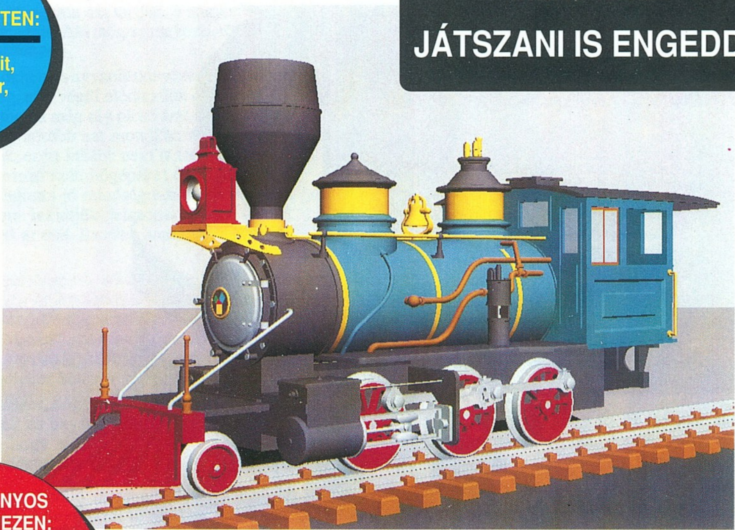
MIKROSZÁMÍTÓGÉP MAGAZIN MÁGNESLEMEZ MELLÉKLETTEL

AZ EXTRA JÁTÉKMELLÉKLETEN:

Arkega, Egasolit, Quinta, Ranger, Solitile, Zone

A HÓNAP TÉMÁJA:

JÁTSZANI IS ENGEDD...



A HAGYOMÁNYOS MÁGNESLEMEZEN:

A billentyűzet munkaterületei C-ben
Ajándék lusta játékosoknak
Moduláljuk a klaviatúrát!
A GET osztály használata
Fej vagy írás?

Magyarországon is Calamus

A Pascal Saga

Dokumentáció óh!

Munkában a szoftverrendőrség

Anyanyelvi programnyelv kezdőknek

Egy kis „daktiloszkópia”

A két konkurens antivírus

A „Unix szelleme” járta be Budapestet

SolarSoft lemezkalauz — hazai műhelyekből

DUPLA MÁGNESLEMEZ MELLÉKLET

IBM. Egy kapcsolat kezdete

Ami sokaknak a kapcsolat befejezését jelenti, az az IBM-nek csak a kezdet. A házasság egy IBM számítógéppel csak az első lépés a hosszútávú és teljeskörű IBM szolgáltatások felé.

Az IBM szolgáltatások sora már a gép megvétele előtt elkezdődik: a tanácsadással. Az IBM a gépek értékesítése után sem hagyja magára a felhasználót, hiszen elvégzi a telepítéssel kapcsolatos munkálatokat is, a kábelezéstől, a szünetmentes áramellátás biztosításán keresztül egészen a gépterem kialakításáig. Sőt, az IBM – mint fővállalkozó – kulcsrakész rendszerek kiépítését is felelősséggel vállalja saját és bármely más gyártó termékei felhasználásával. A rendszer működtetéséhez testreszabott alkalmazási szoftvereket készíttet, illetve adaptál. Hazai és külföldi tanfolyamokat tart.

Nemzetközileg képzett szakemberek, saját alkatrészraktár és a gyorsaság biztosítja a gépek zökkenőmentes működését. Az IBM szolgáltatások közül is kiemelkedik a távkarbantartás, amely a hibák on-line felismerésére és javítására ad lehetőséget.

Az IBM Információs hálózatán keresztül a felhasználók közvetlen kapcsolatot teremthetnek akár hazai, akár külföldi partnereikkel, adatbankkal. Így az üzenetek és információk cseréje számítógéppel történik.

Egy IBM számítógép megvásárlásával tehát a felhasználó részesévé válik a komplex IBM szolgáltatásoknak. És ezek a szolgáltatások jelentik azt a minőséget, amitől az IBM mindenhol a világon IBM.

IBM



GGK

INFORMÁCIÓKÉRÉS: 04 ▲

A minőség megéri az árát!

Amikor elkezdtük, nemcsak mi, hanem a számítógép-vásárlók többsége is kezdő volt. Valamennyien elkövettük azt a hibát, hogy elhittük: érdemes a legolcsóbb, még éppen működő gépekkel foglalkozni. Az évek során azonban rengeteg tapasztalatot szereztünk, és ma már tudjuk: a számítástechnikában sincsenek csodák. A minőségnek és a megbízhatóságnak ára van, s ha ezen spórolunk, az mindkettő rovására megy.



A fejlett piacgazdaságú Nyugaton azt tartják: a nagyon olcsó, bóvli termék a szegények adója. Aki mégis megveszi, rövidesen tapasztalja: rossz üzletet kötött.

A mindenáron való árcsökkentés ugyanis odavezet, hogy egyre gyengébb lesz a minőség, míg végül elérkezünk arra a pontra, amikor ez már nem éri meg a még oly olcsó árat sem. Mi nem akarjuk, hogy ügyfeleink később azt mondják: átvertük őket, egy vackot sóztunk rájuk, ezért inkább nem is forgalmazunk a legalsó árkategóriába tartozó számítógépeket és nyomtatókat. Úgy gondoljuk: egy átlagosan jó minőségre minden vevőnek joga van. Mi ezért nem tekintjük jelszónak, hogy nálunk **a minőség mindig megéri az árát.** Ez több annál, **ez a mi üzleti filozófiánk.**

De ez az ár nem feltétlenül magas — sőt! Először is az **IR** tudja, hogy sokféle igény, feladat és természetesen sokféle pénztárca létezik. Ezért többféle áron, többféle minőséget kínál, az átlagos felhasználónak megfelelő, olcsó „standard" típustól a profiknak szánt, előtesztelt alkatrészekből szerelt, 72 órán át járatott „extra" kategóriáig, amelyhez kétéves garanciát adunk.

De nemcsak ezért lehetséges, hogy az **IR Szervizben** mindenki megtalálja az igényeinek és anyagi lehetőségeinek megfelelő számítógépet. Segíti ezt az is, hogy az **IR** ma már akkora forgalmat bonyolít le, hogy szállítói megadják neki a legjobb vevőnek járó árkedvezményeket. Az **IR** pedig ezeket **a kedvezményeket maradéktalanul átadja** a hazai számítógép-felhasználóknak.

Így lehetséges, hogy egy kiváló minőségű 286-os AT, 40 Mbájtos merevlemezzel, 1 Mbájt RAM-mal, floppyval és nagyfelbontású, színes VGA monitorral **10%-kal olcsóbb az átlag hazai árnál.** Hasonló a helyzet a hordozható számítógépeknél is. De ugyanilyen jók az árai az egyedi igények alapján összeállított konfigurációknak és hálózatoknak is.

És ez még nem minden. **A nyári hónapokra időszakos számítógépvásárt hirdet az IR Szerviz.** Ennek során — konfigurációtól függően — akár **50 000 forintos árkedvezményt** is kapnak a vásárlók. Az oktatási intézmények pedig további nagyon jelentős, **25%-os kedvezményt** élveznek.

Természetesen a választás joga az Öné.
Keressen fel bennünket Budapesten,

IR

IR Szerviz
VII. kerület, Kis Diófa utca 6.
Tel.: 121-3230, 141-0880



írjon levelet, vagy egyszerűen hívjon telefonon, a 121-3230 vagy 141-0880-as számokon, és mi segítünk, hogy kiválassza az Önnek megfelelő számítógépet. **Bármit választ, nyugodt lehet: olyan minőséget kap, amely biztosan megéri az árát!**

INFORMÁCIÓKÉRÉS: 35 ▶

ALAPLAP

Mikroszámítógép magazin mágneslemez melléklettel

Megjelenik havonta

Főszerkesztő: Faklen Pál

Főszerkesztő-helyettes: Varga János

Szerkesztő: Jakab Ágnes

Szerkesztőségi titkár: Sziebig Andrea

A mágneslemez melléklet és a Közkincs szerkesztője: Verebély Pálné

A Lemezkalauz szerkesztője: Vékony Tamás

Szerkesztőség, kiadó és hirdetésszervezés:
XI., Karolina út 17.
Budapest 1251
Telefon:185-2192, 185-2421, 186-9644, 166-2111
Fax: 185-2221

Felelős kiadó:
Sebestyén Ilona igazgató
Cédrus Informatikai Rt.

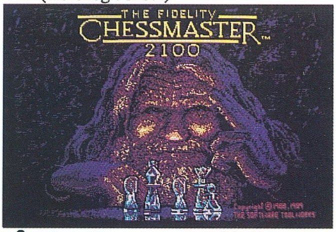
Nyomdai előkészítés:
Tipoprint Kft., Budapest
Nyomtatás:
Zalai Nyomda, Zalaegerszeg
Felelős vezető: Galla József

Terjeszti a Magyar Posta. Előfizethető a hírlapkézbesítő postahivataloknál és a Posta Hírlapelőfizetési és Lapellátási Irodájánál (XIII., Lehel u. 10/a, Budapest 1900), vagy átutalással a 215-96162 pénzforgalmi számra.
Példányonkénti ár: 196 Ft
Évi előfizetési díj: 2 352 Ft

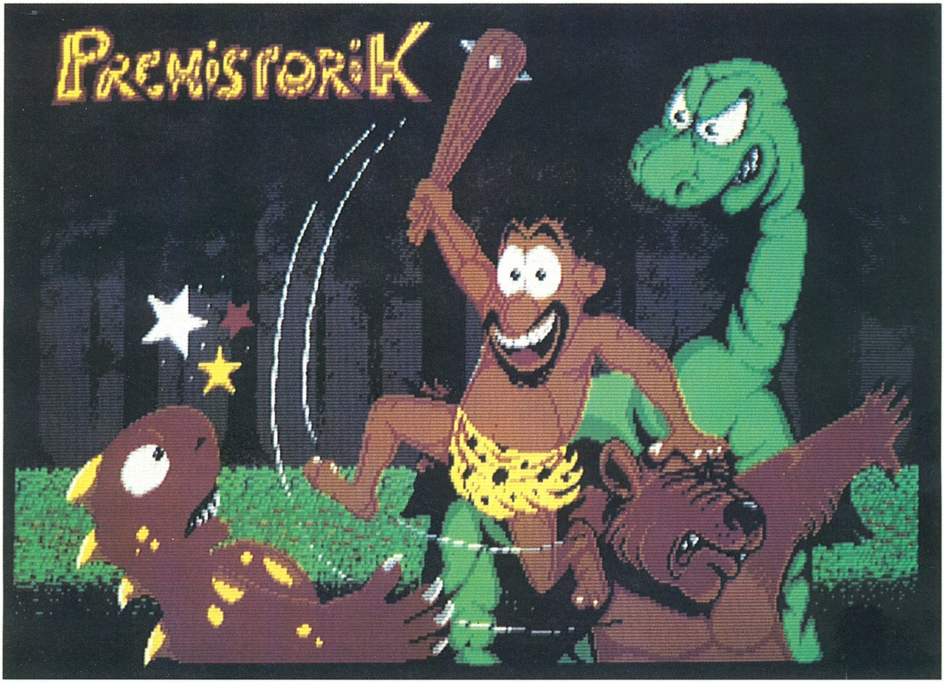
Külföldre terjeszti a Kultúra, Pf. 149, Budapest 1389

HU ISSN 0865-9788

A HÓNAP TÉMÁJA: JÁTSZANI IS ENGEDD...



Címlapképünkön a Pytha cég CAD-grafikája



Homo ludens computeris

Vajon gondolta-e Konrad Zuse valamelyik ősmasinájának készítésekor, hogy szerkezetének kései utódait a hétköznapi életben hol-mindenütt fogják használni? Hogy a kecskeszakállas-szemüveges matematikaprofesszorok exkluzív munkaeszköze bekerül gyermekeink játékai közé is? Aligha hihetjük... mint ahogy Stephensonról sem feltételezhető, hogy előre látta a svájcióra-finomságú elektromos játékvonatokat.

A jövőbe látva nekik is feltűnt volna, hogy az Amiga 3000 vagy a Märklin kisvasút NEM A GYEREKEK JÁTÉKA!

Mivel játszik az apró gyerek? Egyetlen „hanyattesős" macival, Lencsi babával, ad abszurdum színes kavicsokkal. Egy szék pedig bőven elég, hogy vonatosdit játsszon. Fantáziája még oly élénk, hogy azt a valóság bármilyen egyszerű darabkája meg tudja mozgatni, nincs szüksége olyasmire, amit csak felnőttebbé válva tud igazán élvezni.

Vannak apukák, akik őszintén beismerik, hogy a terepasztalt nem a kölyöknek építik. A repülőgép-szimulátor, a kalandprogram, a „virtual reality" valójában a felnőttek játéka. Kinek is szánják a PC-re írt játékprogramokat? A gazdagabb családok elkényeztetett csemetéinek, akiknek 486, 35 MHz, 100 MB, VGA van a kisasztalán? Árulkodó jelként jobbfajta játékprogramokban ott van a „boss key", amire ugyebár nem a főnök gépébe való soron kívüli bejutás lehetősége van kötve, hanem vizuális veszélyhelyzet esetén a képernyőre varázsolható álszövegszerkesztő vagy nagyon tudományosnak látszó táblázat. Mert bizony ezekkel a programokkal a jól szituált apukák a hivatalban, munkaidő alatt is játszanak.

A kisgyermekek játéka — akárcsak az állatkölyköké — felkészülés a felnőttkorra, az igazi szerepekre. Legjobb játék az utánzás: a papás-mamás, a doktor bácsis... De miért játszik a felnőtt? Újabb szerepekre készülne? Talán. Egy menet Pac-Man esetleg jó ujjgyakorlatnak számít egy-egy sűrűbb pull-down menüben való mozgáshoz. A stressz feloldásáért? Ez nem mindig mondható el. A Maniac Mansion beszerzése után napokig nem aludtam, csak töprengtem, és reszketve újra meg újra nekivágtam... De az nem jutott eszembe, hogy abbahagyjam! Vagy az elfojtott vágyak felszínre hozásáért? Mivel nagy valószínűséggel már nem lesz alkalmam az életben sugárhajtású vadászgépet vezetni?

A kérdést, hogy miért játszik a homo ludens, a pszichológusokra és a szociológusokra hagyjuk. A hónap témája mostani számunkban egyszerűen arról szól, hogy a számítógép nemcsak rabszolgánk, hanem játszótársunk is, játék nélküli tudásunk pedig sokkal kevesebbet érne... Tehát, kedves ismeretlen főnök, játszani is engedd!

Zoltai Péter

SimEarth: borzongatóan monumentális

Egy isteni játék

**Az amerikai Maxis szoftverház SimEarth nevű szimulációs programja nem kevesebbet ígér, mint azt, hogy otthon (esetleg a munkahelyünkön?), a számítógép monitorja mellett ülve újrateremthetjük a Földet.
A program szerzői már korábban letették névjegyüket az asztalra a SimCity nevű városmodellező programmal, amelyet pár hónappal ezelőtt az Alaplapban bemutattunk.**

A SimEarth mottója egy angol szójáték: „Is this a random world or did you planet?" A filozófikus kérdés magyarul annyit tesz: Vajon véletlenen alapszik-e a világ, vagy te magad tervezted? (Planet = bolygó, amelynek kiejtése megegyezik a tervez ige „plan it" alakjáéval.)

A SimEarth egy bolygószimulátor, egy planéta élethű modellje. Játék, játékos oktatási segédeszköz, élvezetes időtöltés. Ha a SimEarth-szel játszunk, ne csupán a nyerésre törekedjünk. Kísérletezzünk! Próbálkozzunk! Ne idegenkedjünk a szokatlan megoldásoktól. Már csak a földtörténeti kaland kedvéért is!

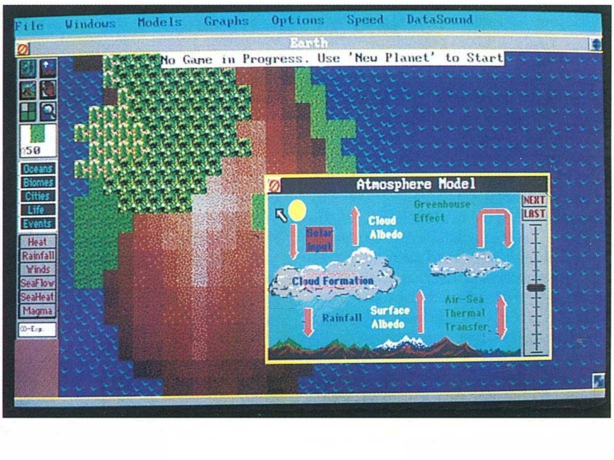
A program több „félkész" bolygót is tartalmaz, de ízlésünknek megfelelően mi magunk is tervezhetünk újabbakat a „kozmikus semmiből". A SimEarth James Lovelock természettudós Gaia elméletén alapul, hogy egy bolygót és az azon kifejlődött életet tekintsük egyetlen integráns egésznek, ne különkülön tanulmányozzuk az élőt és az élettelent. A SimEarth tehát az egészet vizsgálja: a bolygón található életformákat, az időjárást, a légkört, a földfelszínt, a bolygó belsejét. Ezek természetesen folyamatosan hatnak egymásra, és kölcsönösen befolyásolják egymás állagát, állapotát, jellemzőit. A Gaia elmélet szerint — amely ugyan vita tárgya, és a tudományban még nem általánosan elfogadott — egy bolygó nem más, mint bonyolult soktényezős önszabályozó rendszer.

A SimEarth kétféle üzemmódban játszható: versenyszerűen és kísérletezgető formában. A verseny azt jelenti, hogy megadott mennyiségű energiával kell gazdálkodnunk, ami tehát véges, míg a másikban az energia korlátlanul rendelkezésünkre áll, akár pocsékolhatjuk is. A játék erősségi fokozatai is az egyre kevesebb induló energián alapulnak. Néhány szó az időkorlátokról: mint a valóságban is, fokozatosan növekszik a Nap sugárzásának intenzitása, ami végül a Föld pusztulásához vezet. 10 milliárd évünk van a Föld elhagyására, űrhajók segítségével. Négy alapvető időszakot különböztetünk meg egy bolygó fejlődésében: geológiai, evolúciós, civilizált és ipari-technológiai korszakot. Ezeken belül a szimuláció sebessége eltérő. Az újabb időszakokban relatíve „lelassul" az idő a geológiai kor óriási időléptékéhez képest.

A lehetséges életformák a kezdetleges egysejtű mikrobáktól a meleg vízű tengeröblökben egészen az értelemmel rendelkező lényekig (az emberiség megjelenéséig) terjednek. 15-féle osztály létezik a SimEarth-ben, ezek is egyenként 16 fajra oszthatók, azaz egyidejűleg összesen 240-féle életforma tengetheti életét, de az adott körülmények között természetesen nem mindegyik képes a továbbfejlődésre és túlélésre.

A SimEarth civilizációknak 7 fokozatát különböztethetjük meg: kőkorszak, bronzkorszak, vaskorszak, iparosodás, atomkorszak, informatikai korszak és a nanotechnika korszaka. Ezek közül egyidejűleg több is megtalálható lehet a bolygó különböző fejlettségű részein, miként ma, a valódi Földön is élnek még primitív indián törzsek az Amazonas rejtett dzsungeleiben, ahol a tüzet már ismerik, de még nem tudják hasznosítani, miközben a kaliforniai Szilícium-völgyben a SimEarthhöz hasonló szimulációs programokon törik a fejüket a pihent agyú programozók.

A program oktató és magyarázó részeket is tartalmaz. A nem teljesen ismert fogalmak közérthető magyarázatát bármikor megnézhetjük a szószedeten. A program kezelése Microsoft Windows-szerű grafikus ablaktechnikán és legördülő redőnymenükön alapszik. Az egyes ablakokat eltüntethetjük, összecsukhatjuk, kinyithatjuk, elmozgathatjuk, átméretezhetjük. A világűrből is ráláthatunk kreatúránkra, ekkor



a lassan forgó bolygó tárul elénk, s rámutatással tetszés szerinti pontján landolhatunk. Még a bolygó belső felépítését is tanulmányozhatjuk, sőt megváltoztathatjuk, miként beavatkozhatunk a bolygó kinetikai paramétereibe is: a forgás sebességébe, a szimmetriatengely irányába stb...

A SimEarth alaprendszere mindössze néhány szabály, ezek vezérlik az emberi beavatkozásoktól megzavart önszabályozó rendszer válaszait, reakcióit. A globális tényezők között van kémiai, geológiai, biológiai és emberi.

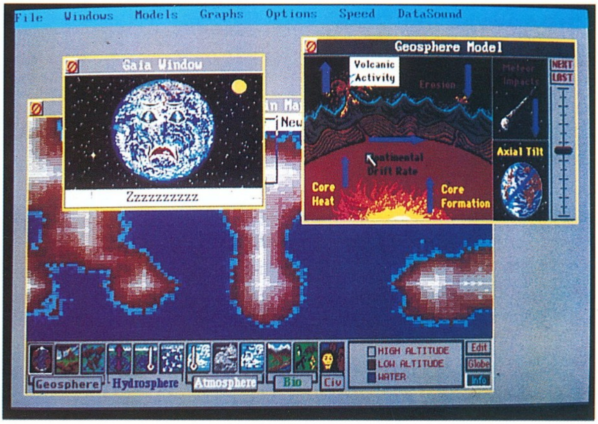
A beavatkozási lehetőségek, mint eszközök, a következők: hegy- és vízrajzi viszonyok módosítása, tengerszint feletti magasság megváltoztatása. Az elemi csapások: hurrikán, szökőár, meteorbecsapódás, vulkanikus működés, nukleáris katasztrófa, tűzvész, földrengés, pusztító járvány. A SimCity-ben még csak utakat, vasutat, házakat, erőműveket építhettünk, itt már hatalmas természeti erőkkel játszadozhatunk.

Az evolúciós láncban egymás után következő különböző életformákat helyezhetünk el a bolygó megfelelő pontjain, az intelligencia fokát növelhetjük. Ha ügyesek vagyunk, beavatkozásaink a fejlődés útjára terelik a bolygót, ellenkező esetben hanyatlását okozzuk, eltüntetve róla az élet fizikai lehetőségét biztosító tényezőket.

A „teremtő munkában" kiváló térképek és szemléletes grafikonok segítenek, miként azt sokan már a SimCity-ben is megszokhatták. A SimEarth természetesen nem tökéletes kópiája egy bolygónak, csupán modellje, vázlata — csodálatos vonalvezetéssel. Uralkodó széljárások, tengeráramlások tanulmányozhatók rajta. A levegő pillanatnyi összetétele (nitrogén, oxigén, szén-dioxid, vízgőz stb.) bármikor a képernyőre kérhető. Ha kurzorként a kézinagyító (lupe) szimbólumot választjuk, akkor a bolygó tetszőleges pontjának pillanatnyi biómáját vizsgálhatjuk, s a program elénk tárja, hogy az adott helyen milyenek az időjárási viszonyok (eső mennyisége, átlagos középhőmérséklet, széljárás), mekkora a tengerszint feletti magasság, a földkéreg vastagsága, melyek a növényzet és az állatvilág jellemző egyedei.

Négy alapmodellen keresztül tudjuk a bolygó leglényegesebb globális paramétereit megváltoztatni:

— A geoszféra modellben többek között befolyásolni tudjuk a vulkáni működés aktivitását, a bolygó belső hőmérsékletét, a talajeróziót, a tengelyelhajlást, a meteorok becsapódási valószínűségét.



— A légköri modellben módosítható a napsugárzás intenzitása, a felhőképződés, az esőzés gyakorisága, a felszíni visszaverődés, a tenger és a szárazföld közötti hőátadási tényező, a melegházhatás, a felhők napfényvisszaverő képessége.

— A bioszferikus modellben változtatható az élőlények melegtűrése, genetikus fejlődési sebessége, szaporasága, mutációs sebessége és a növényzet széndioxid-felhasználása.

— A civilizációs modellen belül az energiagazdálkodás szerkezetét befolyásolhatjuk: a bioenergia, a nap- és szélenergia, a hidro- és geoenergia, a kőolaj és a földgáz, valamint a nukleáris energiaforrások arányát és mértékét. A különböző szakterületek energiafelhasználásával a népesség „béketűrését", illetve agresszivitását, továbbá innovativitását, a tudományokhoz, művészetekhez való vonzódását is manipulálhatjuk. A gyógyítás, a mezőgazdaság, az ipar vagy éppen a filozófia területén munkálkodó emberek számát pedig növelhetjük vagy csökkenthetjük.

Kérhetünk a SimEarthtől ideiglenes helyzetjelentést is. Ilyenkor értékeli eddigi munkálkodásunkat, és kijelöli a legközelebbi földtörténeti lépcsőfokot, amelyet menedzselnünk kell.

Nyolc beépített alapmodellen kezdhetjük a vizsgálódást:

1. Egy teljesen szűz bolygó, ahol még víz és légkör sincs.
2. Az Aquarium nevű ősóceán.
3. A kőkorszak, amelyből ki kell mozdítani a civilizációt.
4. Földanyánk a kambriumi korszakban.
5. A Föld napjainkban (1990).
6. A Mars, ahol növényzetet kell létrehozni.
7. Vénusz.
8. A Gaia-hipotézis.

A SimEarth a Váci utcai FLOPPYLAND szaküzletben 8600 Ft-ért kapható. Magánszemélyeknek árkedvezményként az áfát elengedik.

Jó hír a VGA- és EGA-, valamint Hercules-monitorral rendelkezőknek, hogy a színpompás program a PC-ben található kártya lehető legnagyobb pont- és színfelbontásával indul, s futtatásához csak 640 kilobájt alapmemória és merevlemez szükséges. Az AdLib vagy SoundBlaster hangkártyával rendelkező gépek tulajdonosai még rendkívüli hanghatásokat is élvezhetnek: az ősóceán vagy a fortyogó izzó anyag zajait! A program egérrel is kiválóan vezérelhető, grafikus nyomtatóval pedig megörökíthetjük a földtörténet izgalmasabb állomásait, amelyeket mi magunk hoztunk létre a játék folyamán.

A SimEarth program használatakor a teremtés magasztos érzése lehet úrrá rajtunk. Hogy ez milyen a többnyire pusztításra, lövöldözésre, erőszakra idomító játékokhoz (és filmekhez) szokott generáció számára? Mindenkinek magának kell kipróbálnia! Teremtsünk magunknak egy saját világot, ahol minden a mi elképzelésünk szerint él és virul, és ahol mi vagyunk a mindenható hatalom. Mindenkinek őszintén ajánlhatjuk ezt a borzongatóan monumentális szórakozást, amely egyetlen, de találó szóval megfogalmazva: isteni!

Herczeg József

Chessmaster 2100

Több, mint sakkpartner

A Chessmaster 2000 régóta közkézen forog Magyarországon is. A gyengének korábban sem mondható sakkprogram új változata, a Fidelity Chessmaster 2100 tovább tökéletesedett.

A Software Toolworks által forgalmazott Chessmaster 2100 impozáns csomagolású dobozában a rövid kezelői útmutatón kívül egy vastagabb füzet szólt a sakknak, mint szellemi sportágnak a múltjáról, jelenéről és jövőjéről. Az eredeti lemezekről semmilyen általunk ismert eljárással nem tudtunk biztonsági kópiát készíteni. A gyári lemezről merevlemezre installált program viszont már csak képernyőlementő eljárásainknak állt makacsul ellent — maradt tehát a hagyományos fotózás.

A Chessmaster 2100 (a továbbiakban CM) 654 kilobájtnyi helyet foglal el a merevlemezen, ami egy ilyen tudású programnál teljesen indokolt. Grafikus kártyával rendelkező gépeken futtatható (CGA, Hercules, EGA- és VGA-felbontású monitorral is). A háttér és a figurák színösszeállítását magunk dönthetjük el a felkínált lehetőségek közül.

A megjelenítés háromféle lehet. A háromdimenziós kép szép ugyan, de ember legyen a talpán, aki át tudja tekinteni az állást. (Aki igen, az már vakon is játszhat, amire szintén van mód, bármelyik fél vagy mindkettő figuráit eltüntetve.) A kétdimenziós változat követhető a legjobban, ami nagyjából megfelel a sakkban elterjedt fali demonstrációs táblának, s kérhetünk hozzá a képernyőre külön ablakokat is, amelyekben a CM a játékidőn és a legutóbbi lépéseken kívül mutatja az addig leütött figurákat, a játszma kezdete óta megtett lépéseket, sőt az általa éppen mérlegelt lehetőségeket is — lépéserősséggel együtt.

A játék vezérelhető a klaviatúráról (akár iránybillentyűkkel, akár a koordináták beírásával), botkormánnyal vagy —ami a legkényelmesebb – egérrel. A CM nemcsak ismeri és alkalmazza a sakkjáték összes lehetőségét —mint például a lépés közbeni ütést (en passant), a sáncolást vagy az átváltozást —, hanem oktatja is. Külön „tutorial" üzemmódjában mutatja a lehetséges lépéseket, de beállíthatjuk úgy is, hogy megjelölje az ütésben lévő figurákat.

A CM számos funkciója lehetővé teszi, hogy különböző szinten álló játékosoknak megfelelő partnere legyen. Newcomer (kezdő) stílusban a program mindig csak egy lépéssel néz előre, és csak addig „gondolkodik", amíg rajta a sor, hogy lépjen, ezért ilyenkor érthető módon sokkal könnyebben megverhető. (Akinek egy kis sikerélményre van éppen szüksége, válassza ezt a módot.)

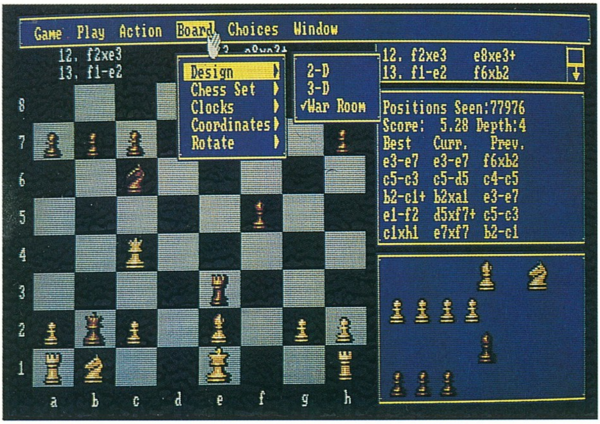
Intermediate (középfokú) szinten a gép szintén csak a saját idejét használja fel az elemzésre, de már több lépéssel előre néz. Véleményem szerint ez a géppel szemben nem igazán korrekt, hiszen a valódi sakkban sincs kikapcsolva az ellenfél agya, amikor éppen mi vagyunk soron.

Legerősebben játszik a program az Advanced (haladó) és a Championship Play (versenyjátszma) beállításnál. A CM útmutatója szerint a program 2353 Élő-pontszámmal egyenértékű játékerejű, ami mesteri fokozatnak felel meg. Versenybeállításnál természetesen tartani kell magunkat a játékszabályokhoz, hogy egy már megtett lépést nem lehet visszavenni, a megérintett figurával kötelező lépni — és hogy ellenfelünk az óra is.

Akinek unalmas (vagy lelkileg megterhelő), hogy a program a Best opcióval következetesen a legjobbat lépi, az választhatja a Normal, tehát közepes módot, de legváltozatosabb a Coffeehouse (kávéházi) stílus, amikor a CM szeszélyesen hol nagyon erőseket lép, hol pedig meglepően egyszerű szituációkat „elnéz". Ha bajban vagyunk, a korábbi lépéseket egyenként visszavonhatjuk, és a játszma tetszés szerinti korábbi helyzetére visszaállva folytathatjuk — most már jól!

Az erős játékstíluson belül is tudjuk befolyásolni a CM játékerejét, mégpedig az egy-egy lépésre vagy az egész játszmára fordítható idő szabályzásával, illetve az lépésmélység meghatározásával. Fix lépésidőnek kérhetünk 2, 5, 10, 15, 30, 45, 60 … 600 másodpercet, de beírhatunk egyéb értékeket is. Másik szintbeállítási módnál meghatározhatjuk a játszma teljes időtartamát 1, 5, 10, 30, 60 percben, vagy tetszőle-





gesen, tehát kiválóan tudunk a számítógéppel „snellezni" is (villámsakk). Ugyancsak időkorlátozási mód, hogy hány lépést kell megadott időn belül megtenni (versenysakkban 40 lépés 120 perc alatt), és aki azt nem teljesíti, automatikusan veszít. A játékerőt módosítja a lépésmélység meghatározása is, hogy a gép hány lépéssel előre számolhatja ki és értékelheti a lehetséges helyzeteket (1, 2, 3, 4, 5 vagy tetszőleges).

Kérhetünk tippeket is a CM-től, s ő gépies becsületességgel mindig az adott fokozatban általa kiszámítható legjobb húzásokat javasolja nekünk. Ha a CM vesztésre áll, ezt azonnal beismeri, és kiírja, hogy felülkerekedtünk. Külön élmény (ha nem is hifi), amikor a lehetséges hangvisszajelzések közül a beszédet választjuk. Ilyenkor a CM ízes angolsággal kommentálja lépéseinket, az ütéseket, a pillanatnyi állást. Érzékeny fülűek ennek használatától azonban óvakodjanak, különösen ha munkahelyükön akarnak sakkozni, és a főnökük nem szeret sakkozni. (Javasoljuk továbbá a Ctrl-Z billentyűt.)

Ha elemezni szeretnénk egy játszmát, és a lehető legjobb lépést szeretnénk kicsikarni a gépből, akkor válasszuk az Infinite (végtelen) időbeállítást. (Különösen ajánljuk sportszerűtlenül játszó levelezési sakkozóknak!) A CM nemcsak ellenünk képes játszani. Felkérhetjük arra, hogy önmagában játsszék, vagy hogy sakktáblát nyújtson két sakkozónak, s mindössze az ő játékukat, a szabályok betartását felügyelje.

Egy adott állást bármikor lemezre menthetünk, s azt később visszatölthetjük, így egy sokáig elnyúló partit függőben hagyhatunk, és másnap folytathatjuk. A játszmák lépései nyomtatóra küldhetők, nem kell azokat kézzel jegyezni, sőt a pillanatnyi állás is kinyomtatható Epson-kompatibilis mátrixnyomtatóval.

A CM-nek betáplálhatunk sakkfeladványokat. Tapasztalatunk szerint ezek megoldásában is jobb, mint elődje, a Chessmaster 2000. Az állások egyszerűen felrakhatók (az egérrel szinte élvezet a figurákat rakosgatni vagy eltüntetni), és csak a nagyon trükkös soklépésűekkel lehet neki gondot okozni.

A csomagban 110 klasszikus sakkparti található, szöveges elemzésekkel. Híres sakkozók, többek között Aljehin, Capablanca, Szmiszlov, Fisher, Tal, Szpasszkij, Ribli, Karpov, Kaszparov jól sikerült játszmáit csodálhatjuk meg, és játszathatjuk újra.

Nagyon hasznos szolgáltatás a CM-ben a kiterjedt megnyitási adattár. A szerzők több mint 151 ezer állást kódoltak ebbe az egy adatbázisba, egyes ismertebb megnyitásoknál 15-20 lépésig is. Ha az Opening Book opciót bekapcsoljuk, a CM mindig kijelzi, éppen melyik ismert megnyitási logika és algoritmus szerint folyik a játék. Mód van a kiválasztott megnyitások elsajátítására is (sánciskola), amikor a CM szinte a kezünket fogva segít begyakorolni a lépéssorozatot.

Felkérhetjük a programot, hogy folyamatosan mérje játékerőnket. A kijelzett Élő-pontszám körülbelül 80-100 játszma után tekinthető igazán mérvadónak, de azt azért ne várjuk el, hogy a Chessmaster által megítélt pontszám alapján a FIDE soron következő kongresszusán előterjeszthetnek bennünket nemzetközi nagymesternek.

Herczeg József

Mahjongg-ból Solitile

Klasszikus — modern átiratban

A hagyományos játékok közül különösen jól adaptálhatók számítógépre azok, amelyekben az ellenfél nem „Valaki", hanem maga a Játék, a Véletlen. Az unalomŭző kirakós kártyajátékok változatos családját pasziánsz néven emlegetjük, de még kifejezőbb a latin solitarius (magányos) szóból származó szoliter elnevezés. Ezt használja az Alaplap e havi számának ráadásként adott mágneslemez játékmellékletén található Solitile is. A „tile" angol szóvég jelentése pedig cserép, kőlap, játékzseton, amivel az ilyen típusú játékokat eredetileg — már az ókorban is — játszották.

A képernyőn is nagy karriert befutott kínai Mahjongg (vagy Shanghai) játék eredeti ábrái, szimbólumai az európai szemnek kicsit szokatlanok, s talán ezért határozták el a Solitile készítői, hogy „szentségtörően" profanizálják, a nyugati civilizáció modern hétköznapi tárgyaival és jelzéseivel helyettesítik azokat. Azt is egyhangúnak találhatták, hogy a zsetonok mindig ugyanabban az alakzatban legyenek, ezért eleve 9 különböző forma közül lehet választani, és magunk is szerkeszthetünk újakat.

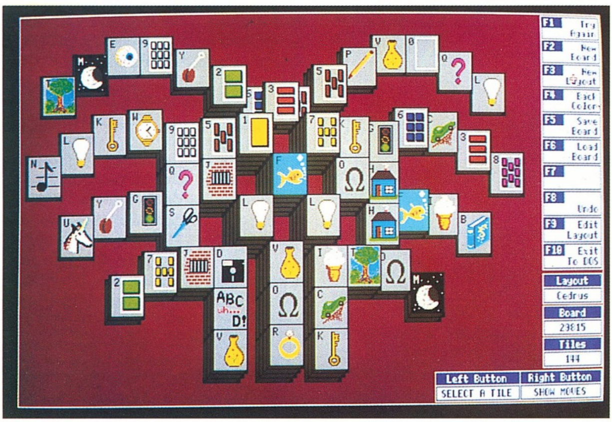
Az ősi játék alapszabályain viszont nem változtattak. A teljesen lebontandó kupacban 36 különböző ábrájú cserép mindegyike 4-4 példányban fordul elő, tehát összesen 144 zsetonnak kell elfogynia oly módon, hogy mindig rákattintunk két-két egyformára. Azok viszont csak akkor távoznak, ha egyáltalán nincsenek takarva, és bal vagy jobb oldalról nem blokkolja őket azonos szinten lévő másik „kő".

A program a zsetonok kupacon belüli elhelyezkedése szerint összesen 65536 (2^{16}) variációt számozottan tárol, s annak alapján mindegyik felállást bármikor tálalja (például: solitile –b 27831), egyébként pedig véletlenszerűen választ a lehetőségek közül. A Solitile természetesen az egérrel kezelve igazán kényelmes, de a billentyűzettel is könnyen lehet vele bánni. (Az egér bal odali gombjának a Home, a jobb oldalinak a PgUp billentyű felel meg.) Ha lankad a figyelmünk, többféle segítséget kérhetünk: felvillanthatjuk, hogy van-e szabad megfelelője annak a zsetonnak, amelyikre rákattintottunk, megmutatja a hozzáférhető párokat, visszaléphetünk a zsákutcának bizonyult útról, hogy másik ágon mehessünk tovább stb.

A program egyik gyengéje — amit a szerző a kísérő dokumentációs fájlban korrektül be is vall —, hogy szerkesztési módban az egér használata nehézségekbe ütközik, ezért jobb, ha ilyenkor teljesen a billentyűkre hagyatkozunk. A program EGA/VGA hardvert igényel, de reméljük, hogy ez ma már egyre kevesebb helyen okoz gondot.

Akik először játszanak a Solitile programmal (vagy más, Mahjongg-típusú játékkal), tapasztalják majd, hogy a kövek felszedési sorrendjén változtatva — tehát más taktika, másik út választásával — megoldhatók előzőleg megoldhatatlannak bizonyult feladatok is. De ne gondolják, hogy mindig! Az azonos ábrájú zsetonok számos variációban kerülhetnek olyan takarásba vagy lehetnek úgy leblokkolva, hogy egyetlen útvonal sem vezet kiszabadításukhoz. Az esetek többségében azonban győzhetünk.

Faklen Pál



A funkcióbillentyűk játék üzemmódban:
F1 Újrajátszás
F2 Új jászma
F3 Másik alakzat
F4 Háttérszín változtatása
F5 Játszma elmentése
F6 Játszma betöltése
F8 Visszaléptetés
F9 Alakzat szerkesztése
F10 Kilépés a játékból

A funkcióbillentyők szerkesztési módban:
F1 A tábla kiürítése
F2 Új alakzat betöltése
F3 Az aktuális alakzat törlése
F4 Háttérszín módosítása
F5 Aktuális alakzat névváltoztatása
F7 Változtatások elmentése
F9 Vissza a játék módba (mentéssel)
F10 Kilépés a játékból

Maniac Mansion, avagy egy újfajta szerepjáték

Spielberg a monitoron

A szöveges kalandjátékok (adventure) talán a legrégebbi számítógépes „műfajba" tartoznak. Még sehol nem voltak az interaktív személyi számítógépek, mikor egyik programozó barátom áradozva mesélt egy „óóriááási" kalandjátékról, melyet az operátorok hónapok óta játszanak éjszaka a konzolírógépen, a végkifejlet halvány reménye nélkül. Valóban, ehhez az „okos, de vak ember és az ő látó, de buta szolgája" típusú szerepjátékhoz egy tetszőleges kiírószerkezeten (teletype, CRT stb.) és egy billentyűzeten kívül semmilyen külön hardverre nincs szükség.

A történet egy húsz évvel ezelőtti, fülledt nyáréjszakán kezdődik. Az éji csendben csak a tücskök ciripelése hallatszik, amikor hirtelen egy üstökös szeli át dobhártyaszaggató robajjal az égboltot. Doktor Fred városszéli dácsája mellett csapódik be, amit néhány hétre rá már mindenki csak Bolondok Házának nevez az amerikai kisvárosban. Fred doki, felesége, Edna nővér és fiuk, Lökött Ed olyannyira visszavonultan élnek, hogy még bevásárolni sem járnak a városba. A kerítésen tábla figyelmezteti a hívatlan látogatókat: „A belépők rettenetesen megcsonkíttatnak!" — nem is közelít hát a házhoz senki. Az embereknek azonban piszkálja a csőrét a rejtélyes villa, s mindenféle mendemondák kelnek lábra. Egyesek tudni vélik, hogy Fred doki a házában rejtegeti rég megboldogult kuzinja, Ted földi maradványait, mások borzalmas orvosi kísérletekről, mi több, viviszekcióról suttognak... Ezen az éjszakán végre történik valami! A csinos Sandy eltűnik, barátja, Dave azonban még látja, hogy a dilis doki a Maniac Mansion felé vonszolja. Sebtében mentőakciót szervez, két barátjával Sandy kiszabadítására indul. Ez az a pillanat, amikor deus ex machinaként, az események irányítását a számítógép előtt ülő játékos veszi át. A gép röviden bemutatja Dave barátait, mind jópofa srác vagy lány, s mindről kiderül: másmás egyéniség, eltérő jó tulajdonságokkal és különböző hátrányokkal. Van közöttük punkzenész, fotóművész, s akad a fizikának avatott tudora is. Mindannyian sajnos nem mehetnek, de annyit megsúgok, a szöszke Sandyt bármilyen felállásban ki lehet szabadítani.

Az adventure játékok sohasem vesztettek népszerűségükből, sőt a személyi számítógépek lehetőségei (színes grafika, hang) egy-egy kalandozás erejéig a kisebb képzelőtehetségű játékosokat is a gép mellé ültették. Látványosan bemutatható a táj egy részlete, a gonosz manó, s hallani az aranymosók kis patakjának csobogását is...

Kifejezetten a mikrogépeken terjedt el a lövöldözős (arcade) játéktípus, melyekből tizenkettő egy tucat. A szabály egyszerű: lőj mindenre, ami mozog („shoot'em up"). Valamivel szellemesebbek az akció-játékok, ahol szaladgálni kell az ellenség elől, kulcsokatmuníciót felszedni, s persze itt is lehet közben lövöldözni.

Az alkotó elme mindig újabbat és nagyszerűbbet kíván, s a piac törvényei is ezt diktálják. Egyszerűségében nagyszerű az a gondolat, hogy egyesítsünk két játéktípust, mindkettő erős oldalait megtartva. Az akciójátékban nem kell sokat szövegelni, inkább cselekedni kell: „Látlak, gonosz fajzat, s menten le is lőlek!", vagy „Hopp, ez a tárgy jó lesz még valamire, fölveszem". Azt se felejtsük el, hogy az ember információinak kilencven százalékát vizuális úton szerzi.

A szerepjátékok a számtalan logikai („most mit tegyek") és térbeli („most merre menjek") elágazás miatt olyan sok variációt rejtenek, mint egy sakkjátszma — nem akad két egyforma. Kiválasztott figuránk irányításunk szerint mozog, tesz-vesz, „ront vagy javít, de nem henyél". Elmarad a helyszín, a szereplők, a tárgyak leírása, hiszen mindenki látja, hogy a herceg szakálla kék, s hogy hétszobás a lakosztálya. A történések is felgyorsulnak: nem kell megvárni, míg a játékos elolvassa, hogy a vak küklopsz súlyos tárgyakat hajigál, ha nem ugrik félre. Adrenalin-szintünket az izgalom és kíváncsiság egyformán emeli, az élmény hatványozódik.

Az előbb említett Maniac Mansion olyan, mintha Steven Spielberg filmjét figyelné az ember, azzal a nem elhanyagolható különbséggel, hogy most magunk vagyunk a rendezők! Nagyon találó az az újabb „-ware" szóösszetétel, amit egy másik játék előzetesében olvastam: cinemaware! Az erőszakosak betörnek majd Dave-vel, vagy verekedni próbálnak. A lágyszívűek biztosan megkísérlik az ellenfeleket saját pártjukra állítani. Tessék, hölgyek, urak, csak folyvást, folyvást! A filmszerű élményt fokozzák azok a kis számítógép-vezérelte intermezzók, amelyekből megtudhatjuk, mit csinálnak közben a lökött família tagjai, vagy mit forgat zavaros fejében dr. Fred a csinos Sandyvel kapcsolatban. Nem alaptalan a mozi-hasonlat azért sem, mert a játékot az a Lucasfilm Games Ltd. készítette, melynek vezetőjét, George Lucast bizonyára nem kell a mozilátogatóknak külön bemutatni.

Zoltai Péter

Stratégiák harca

Játék — matematika — számítógép

Az emberi tevékenységek közül talán a játék áll legközelebb a matematikához. A legkomolyabb matematikában is rengeteg a játékos elem, és megfordítva, számtalan olyan játék van, amelyben komoly matematikai problémák vannak elrejtve. Mély rokonság van az ismert matematikai struktúrák és a játékok struktúrája között is.

Évszázadokon keresztül jórészt a természettudományok szállították a matematika számára a tisztázandó problémákat, ami kétségtelenül rányomta bélyegét a matematika egész fejlődésére. A XIX. század közepe óta fokozatosan megváltozott a matematikusok felfogása arról, hogy milyen jelenségekben tartják kompetensnek a matematikát. Ma már a nyelvtudománytól a közgazdaságtanig alig találunk olyan tudományágat — a társadalomtudományokat is beleértve — amelyben ne hozott volna forradalminak is bátran nevezhető változásokat a matematikai módszerek — és természetesen a számítógép alkalmazása.

Nem is olyan régen még kinézték maguk közül a „tiszta matematikával" foglalkozó szakemberek a „gépeseket". Igaz, ez még az előtt volt, hogy a számítógép segítségével sikerült például megbirkózni a matematikusoknak egy olyan kemény tétel bebizonyításával, mint az ún. „négyszíntétel". Látszólag ez is csak játék, hiszen mindössze annyit állít, hogy bármilyen térkép kiszínezhető négy különböző színnel úgy, hogy a szomszédos területek színe mindenütt eltérő legyen. Az összes lehetőség logikus végiggondolása azonban, bármennyire hihetetlen, meghaladta az ember számítási kapacitását.

Játékelmélet

Sok matematikai probléma játékos formában is megfogalmazható, sőt, a játék szabályai módosíthatók, szigoríthatók vagy lazíthatók, éppúgy, mint a matematikai modellek előírásai. Ezzel lehántható a problémáról minden felesleges cafrang, elhagyható a „játékból" minden zavaró momentum, a „játékszabályok" sorozatos módosításával pedig megsokszorozható, fokozatosan bonyolítható és variálható a játék.

A játékok struktúráját az egyetemet alighogy elvégző Neumann János kezdte komolyabban — matematikai módszerekkel — vizsgálni, 1928-ban. A fiatal Neumann kutatásaiból kerekedett ki azután a játékelmélet: egy új matematikai diszciplína. Bár a játékelméletből leszűrt eredményeket ma már sokkal inkább az elméleti közgazdaságtan kamatoztatja, és — valljuk be — a játékosságot már inkább csak az elnevezés őrzi, kétségtelen, hogy a játékok elméleti igényű vizsgálata ma sem nélkülözheti a játékelméleti megalapozást.

Neumann János nyomán a játékelméletet úgy foghatjuk fel, mint az optimális stratégiák keresésének elméletét. Maga a „játék" abból keletkezik, hogy minden játékos a maga stratégiáját igyekszik megvalósítani: a játékot az ellentétes érdekeket követő stratégiák összecsapása biztosítja. Úgy is szoktak fogalmazni, hogy a játékelmélet a konfliktusok optimális megoldásának elmélete.

Rendszerint minden játékos arra törekszik, hogy saját veszteségeit minimalizálja, függetlenül attól, hogy milyen stratégiát követ a másik (vagy a többi) játékos; más szavakkal, hogy milyen a többiek ellenjátéka. Ez az ún. „minimax stratégia" — amelynek pontos definícióját és a vele kapcsolatos legfontosabb tételek bizonyítását Neumann János adta meg — a játékosok logikus viselkedését fejezi ki. Amennyiben a játékos birtokában van a teljes információnak, a minimax stratégia előre pontosan meghatározható, és szabályok formájában előírható, hogy az ellenfél milyen lépésére milyen ellenlépéssel kell válaszolni. Ha azonban véletlen elemeket is tartalmaz a játék (kockadobás, kártyahúzás stb.), akkor az optimális stratégia meghatározásához valószínűségelméleti meggondolásokat is figyelembe kell venni. Ilyenkor a játék számunkra optimális lefolyásának csak a valószínűsége biztosítható.

A játékelmélet alapvető problematikája akörül forog, hogy az optimum kritériumait hogyan lehet meghatározni a játékok különböző osztályaira, milyen matematikai sajátosságok jellemzik a különböző játékosztályokat, illetve azokon belül az optimumkritériumok realizálódását, és hogyan lehet ezeket megállapítani.

Az osztályok megkülönböztetésére vonatkozólag a matematikusnak nincs prekoncepciója. Éppen az elemzésből, a játékok matematikai viselkedéséből kell, hogy kiderüljön, melyeket érdemes külön vizsgálni, külön osztályokba sorolni. Amikor például a matematikus „véges, kétszemélyes, zérus összegű, teljes információval és optimális stratégiával rendelkező játékok osztályáról" beszél, akkor ez a következőket jelenti:

1. A játékot két személy játssza.
2. A játék véges számú lépés után befejeződik.
3. Mindig van egy győztes és egy vesztes, azaz a nyereség és a veszteség összege nulla.
4. A játéknak van olyan egyértelmű lefolytatási menete, amely az első lépést megtevő játékosnak — ellenfele tetszőleges lépései ellenére — biztosítja a győzelmet.

„Nem véletlenül..."

Ősi, a világ számos országában jól ismert játék a „kő–papír–olló", bár érdekes módon Magyarországon kevéssé elterjedt. Két játékos játssza, minden segédeszköz nélkül, csak a kezükkel mutatnak vagy „követ" (ökölbe szorított kéz), vagy „papírt" (kinyújtott tenyér), vagy „ollót" (három kinyújtott ujj). A játék lényege az, hogy ütemes karlendítéseket végezve (egy... két... hár...) harmadikra mindkét játékos egyszerre mutatja valamelyik alakzatot. Aki az erősebbet választotta, az nyeri

az adott fordulót. Legerősebb alakzat azonban nincsen, mert a kő erősebb az ollónál (kicsorbítja), de gyöngébb a papírnál (bele lehet csomagolni), a papír erősebb a kőnél, de gyöngébb az ollónál, az olló pedig erősebb a papírnál, de gyöngébb a kőnél. Tíz nyertes fordulóval lehet megnyerni a játékot, feltéve, hogy megvan a 2 pont különbség kettejük között. Ha nincs, addig folytatják, míg ezt el nem érik.

A játékelméletben szokásos ábrázolással így írhatjuk fel a feltételeket kifejező nyereségmátrixot (más elnevezéssel elszámolási mátrixot):

	Kő	Papír	Olló
Kő	0	–1	1
Papír	1	0	–1
Olló	–1	1	0

(A játék kínai változatában egyébként a játék lényege és a nyereségmátrix is ugyanaz, csak náluk mások a „szereplők": ott az ember megeszi a kakast, a kakas a férget, a féreg az embert.)

Játékelméleti elemzéssel megállapítható, hogy az optimális stratégia mindkét játékos számára az, ha mindig véletlenszerűen mutatnak követ, papírt vagy ollót. A valóságos játék azonban más képet mutat. Kiderül ugyanis, hogy az ember képtelen véletlenszerű jelsorozatokat produkálni, és nem az nyer, aki találomra mutogat, hanem aki jobban kiismeri a pillanatnyilag érvényesülő szabályszerűséget az ellenfél játékában, vagyis képes az ellenfél stratégiájának ellenstratégiáját játszani. (Lásd a snóbliról szóló cikket is a 15. oldalon.)

A szuperravaszság

Nézzük csak, milyen pillanatnyi stratégiák vezérelhetik az ellenfél játékát!

1. „Lusta" stratégia

Az ellenfél lusta gondolkozni. „Hár..."-ra is megmarad ugyanabban az „alapállásban", amiben addig volt.

2. „Makacs" statégia

Az ellenfél makacs, és csakazértis ugyanazt mutatja, amit az előző fordulóban mutatott.

3. „Bolond Miska" stratégia

Az ellenfél azt csinálja, amit a mesebeli Bolond Miska: utólag játssza meg azt, amivel az előző fordulóban nyerhetett volna.

4. „Ravasz" stratégia

Arra készül fel, amit szerinte az ellenfél játszani fog, reagálva az utolsó forduló sikeres, sikertelen vagy döntetlen eredményére.

5. „Duplán ravasz" stratégia

Túl akar járni az ellenfél eszén: arra számít, hogy az ellenfél a ravasz stratégiát alkalmazza, de ő még ravaszabb akar lenni. („Ő azt hiszi, hogy én azt hiszem, hogy ő azt hiszi...")

...Itt folytathatnánk tovább is a „triplán ravasz" stratégiával, majd annak ellenstratégiájával, de bármily fájdalmas, a különbözőnek feltételezett stratégiák egyre inkább egymásba folynak. Mindez azonban nem zárja ki, hogy ne lehetne modellezni, sőt számítógéppel elemezni a játszó ember játékmodorát, viselkedését: hogy reagál a nyerésre, hogyan a vesztésre, illetőleg a döntetlen fordulóra, továbbá mennyire tanulékony: mennyire tudja egyrészt rugalmasan váltogatni a különböző stratégiákat, másrészt kiismerni az ellenfél stratégiáját és túljárni az eszén.

Aki néhányszor már megpróbálta, annak számára nyilvánvaló, hogy nem szerencsejátékról, hanem mély pszichológiai töltetű játékról van szó. Egy (de méginkább több) jól megírt számítógépes programmal roppant érdekes kísérleteket lehetne végezni. Meg lehetne oldani vele — természetesen hosszabb játéksorozatok alapján —, hogy típusokba soroljuk és számszerű értékekkel jellemezzük, ha nem is a partner IQ-ját, de agyának elevenségét, rugalmasságát, megfigyelőképességének gyorsaságát. Össze lehetne ereszteni és versenyeztetni lehetne több, különböző koncepciókat megvalósító programot is, egymás intelligenciájának kifürkészésére, értékelésére.

Munkára fel, reménybeli számítógépes pszichológusok! Az Alaplapban többször is közreadunk majd olyan programokat, amelyek nemcsak hogy okosabbnál okosabban játszanak ellenünk, hanem egymással is versengve képesek egyre több hasznos következtetést levonni játékunkról — és rólunk!

Vargha Dénes

Billentyűzetmakró — lustáknak

Bizonyára sokan bosszankodtak már azon, hogy egyes ügyességi játékokban, ha valamit eltévesztünk, egy jóval korábbi helyzetből kell újra elindulni. Milyen jó lenne, ha az editoroknál (pl. Multi-Edit) megszokott módon rögzíthetnénk a lenyomott billentyűk sorrendjét, és amikor szükséges, visszajátszhatnánk azokat. A feladat azonban a játékprogramoknál ennél jóval bonyolultabb: nemcsak a billentyűk sorrendjét, hanem lenyomásuk idejét is rögzíteni kell, és a makro futtatását meg is kell tudni állítani.

Olyan programot írni, amely minden játékprogramnál jól működik, csaknem lehetetlen. A játékprogramok íróinak kedvenc szokása ugyanis, hogy a billentyűzetet közvetlenül kezelik, a BIOS megszakítások megőrzésével ritkán foglalkoznak.

A mágneslemez mellékleten közreadott UMACRO.COM program azoknál a játékoknál jelent segítséget, amelyek a szokásos módon, a 16h szoftver interrupton keresztül kezelik a billentyűzetet.

Az UMACRO.COM memóriarezidens program. Betöltése után használata igen egyszerű. A billentyűlenyomások rögzítését az F9-cel kell kezdeni és az F10-zel lezárni. A visszajátszás az F8-ra indul. A billentyűk és a leütésük közötti szünetek a rögzítéssel megegyeznek. A visszajátszás bármelyik gomb lenyomásával megszakítható. A vezérlő billentyűk természetesen átdefiniálhatók. Ehhez az USETUP.EXE programot kell használni. Az egyszerű menüvezérelt setup kezelése magától értetődő. Itt lehet megnövelni a rögzíthető billentyűk számát is. Ezzel persze a program memórafelhasználása is nő. Az alapértelmezés értéke 100, a lehetséges maximum kb. 5000.

A változtatások természetesen csak az UMACRO.COM legközelebbi betöltése után működnek. Akik a program működésére is kíváncsiak, azok az Assembly nyelvű forrásprogramot is megtalálhatják a mágneslemezen. Ennek felépítése a memóriarezidens programoknál megszokott.

A kódok rögzítésére a puffertől kezdődő memóriaterület szolgál. Az első elem nem használt, így a program egyszerűbben működik.

A setup program Turbo-C-vel fordítható. A program a parancssor beolvasása után megnyitja az UMACRO.COM fájt, és megkeresi benne az azonosítót. Ez után vannak az átírható változók. A változók módosítását a menu( ) eljárás végzi. Ha szükséges, akkor a módosításokat a főprogram átírja az UMACRO.COM fájlban.

Pintér Gábor

Utolsónak lenni

Mennyit számít egy plusz dimenzió!

Új játékok kitalálásának egyik legtermészetesebb módja a szabályok módosítása, a tábla alakjának vagy méreteinek a változtatása, vagy pedig új dimenzióval való bővítése. Így lesz például a jól ismert (és optimális stratégiájának kitanulása után hamarosan megunt) NIM játékból egy teljesen új és érdekes módon matematikailag is eredeti játék.

A NIM játékban az veszít, aki rákényszerül, hogy az utolsó babszemet (gyufaszálat, kavicsot vagy bármi egyebet) elvegye. 100 darabból felváltva vehetnek el a játékosok egyszerre legalább 1-et és legfeljebb 9-et. Ezt a játékot nagyon könnyű „kitanulni", vagyis nagyon könnyű felismerni és alkalmazni a nyerő stratégiát — kár is volna vele a számítógépet terhelni.

Egészen más a helyzet a játék kétdimenziós továbbfejlesztésével.

A játékot ketten játszhatják, tetszőleges méretű téglalap alakú táblán. (Egyszerűség kedvéért elővehetünk egy sakktáblát, és azon jelölhetjük ki a kiválasztott tábla méreteit.) Töltsük ki a kiválasztott téglalapot babszemekkel, mondjuk úgy, hogy a kiválasztott téglalap a tábla bal felső sarkából induljon ki, jobbra és lefelé — ez a megállapodás a képernyőn való könnyű megjelenítés érdekében célszerű. A soron következő játékosnak ki kell választania az i-edik sor k-adik elemét, és levenni a tábláról az ettől jobbra és lefelé eső valamennyi babszemet. A másik játékos az így keletkezett lépcsős alakzatból „harap" le hasonló módon. Az veszít, aki végül kénytelen elvenni a bal felső sarokban lévő „mérgezett" babszemet.

Lássunk néhány egyszerű példát! Egy 3 x 4-es táblán ez a kezdő helyzet:
0)

• • • •
• • • •
• • • •

Világos kezdőlépése legyen például (2, 3). A kapott alakzat:
1a)

• • • •
• •
• •

Sötét válaszlépése mi lehet? Ha (2, 1)-et lép, világos (1, 2)-vel azonnal kikényszeríti a vesztes (1, 1) lépést. Hasonlóan azonnali vesztéshez vezetne sötét számára (1, 2) is, mert erre meg (2, 1) világos válaszlépése. Próbálkozzunk sötét nevében a (2, 2)-vel:
1b)

• • • •
•
•

Világos megint nyerni tud, mert ha (1, 4)-et lépi, a kapott
2a)

• • •
•
•

alakzatból mindenképpen ő kerül ki győztesen. Mi lenne, ha sötét nevében (1, 4)-gyel folytatnánk világos (2, 3) kezdőlépésére? Íme:
1b)

• • •
• •
• •

Sajnos, ez sem jó, mert világos ebből is előállíthatja a következő lépésben (2, 2)-vel a számára nyerést biztosító „sarokalakzatot":
2a)

• • •
•
•

Talán (3, 1) nyeréshez vezet? Próbáljuk csak:
1b)

• •
• •
• •

Világosnak erre is van ellenlépése: (3, 2).
2a)

• •
• •
•

Ebből az alakzatból ugyanis (1, 2) és (2, 1) azonnali vesztéshez vezet, (2, 2)-re pedig
2b)

• •
•
•

(3, 1)-gyel felelne:
3a)

• •
•

megfordítva pedig (3, 1)-re
2b)

• •
• •

(2, 2)-vel:
3a)

• •
•

ami ugyanúgy az ő számára biztosítja a nyerést, mint a fentebb vizsgált „sarokalakzat":

• • •
•

A probléma, gondolom, ennyiből már világosan kiderül: hogyan lehet számítógéppel előállítani az optimális stratégiát biztosító „kitüntetett alakzatokat", amelyeknek elérése biztosítja a játékosnak (megfelelő program esetén az ember ellen játszó számítógépnek), hogy bármit lépjen is az ellenfél, mindig ő legyen az, aki újból és újból kikényszerítheti, hogy neki jussanak a nyerést biztosító alakzatok.

Két hónap múlva visszatérünk a probléma elemzésére. A beküldött anyagokból szívesen közlünk is néhányat, például:

a) Egy mutatós játékprogramot a játék képernyőn való megjelenítésére.

b) Számítógépes programot maximum 8 x 8-as méreten belül valamennyi „kitüntetett alakzat" előállítására, például vektorok formájában: (2, 1), (2, 2, 1), (3, 2) ...

c) Programot a „kitüntetett alakzatok" adatbázisban való tárolására, adott szempontok szerinti előkeresésére és megjelenítésére.

A legjobb programok szerzőit 3-3 szabadon választott Solarsoft shareware programlemezzel jutalmazzuk!

Vargha Dénes

A PC-re írt sakkprogramok forradalma

Az Ördög alszik?

Sokáig a mikroszámítógépekre írt legjobb sakkprogramok sem nem tudták felvenni a versenyt még a közepes játékerejű specializált sakk-számítógépekkel sem. A közelmúltban azonban megjelentek olyan új PC-programok, amelyek komoly kihívást jelentenek a legjobb „célgépeknek” is.

Mi volt az oka annak, hogy a sakk-számítógépek olyan hosszú időn át jobban sakkoztak a PC-s programoknál. (Pedig a sakk-számítógépek programozói is PC-ken fejlesztenek, és utóbb ültetik át művüket.) A fő ok, hogy az egyetlen feladatra „lebutított” (illetve „felokosított”) gépekkel jobb sakkteljesítményt lehetett elérni, mint a sokoldalúságukat ebben az esetben hátrányként hordozó PC-kkel. Ráadásul PC-programok valódi játékerejét meg sem lehet határozni, mert az nagymértékben függ attól, hogy a programot milyen PC-n futtatják. Ugyanaz a program kitűnően játszik egy 80486-os, 33 MHz-es AT-n, és nagyon gyenge ellenfél egy 10 MHz-es turbo XT-n. A fejlesztők végülis arra a következtetésre jutottak, hogy a sakkozás különleges funkcióinak eredményesebb megoldására a PC hardverjét kell tovább fejleszteni.

Azért közben a PC-sakkprogramok is fejlődtek. A sakk-automaták legkiválóbb programozói mintegy mellékesen PC-re is fejlesztettek, de programjaikat nem a sakk-számítógépeket készítő cégek forgalmazták, mert a Mephistokat gyártó müncheni Hegener+Glaser, az amerikai Fidelity, a hongkongi Saitek vagy a NOVAG közül egyik sem óhajtott önmagának versenyt támasztani.

Mégis megszülettek a világbajnok Mephisto brit programozója, Richard Lang műhelyében a Psion programok, a Fidelity (újabban a Saitek) számára dolgozó kaliforniai Spracklen házaspár által készített Sargonok, és nem utolsó sorban a nálunk is jól ismert, népszerű Chessmaster 2000 és 2100, amelyeknek alkotó csoportja Los Angelesben dolgozik, és vezetőjét, David Kittingert ugyancsak a világ legjobbjai között tartják számon. (A Chessmaster 2100-ról lapunk 6–7. oldalán részletesebben is írunk. A szerk.) A legújabb információk szerint hamarosan a Chessmaster 2250-es verzió is a piacra kerül, vagy időközben ez már meg is történt.

Az előtörténetben meg kell még említeni a szinte klasszikusnak mondható Cyrus sorozatot a brit David Levy-től, aki az ICCA (International Computer Chess Association) elnöke, Mark Taylor által vezetett csapatban dolgozik, valamint a nálunk is ismert Colossus verziókat a brit Martin Bryant-tól. Az eddig említett programozók valamennyien itt voltak Budapesten 1983-ban, amikor Magyarország rendezte meg a 3. mikoszámítógépes sakkvilágbajnokságot.

A korábban sikeres vagy népszerű sakkprogramok közé tartozik még a Chessplayer 2150, a Chess Simulator vagy a nem túl erős, de igen látványos Battle Chess.

Talán különös, de a legtöbb program pályafutása általában világbajnokságon vagy más nagy nemzetközi vetélkedőn kezdődik. A még kísérleti stádiumban lévő vagy amatőrök által készített szoftverek rendszerint itt mutatkoznak be. 1990 novemberében Lyonban, a Kasparov–Karpov sakkvilágbajnoki mérkőzéssel párhuzamosan rendezték meg a mikroszámítógépek 10. világbajnokságát, amelyen első ízben kerültek a legjobbak közé kifejezetten PC-re írt programok: a holland Gideon és The King a 3.-4. helyet érte el. Ennél is jobban szerepeltek röviddel utána, idén áprilisban, Vancouverben, a 11. vb-n. Gideon megnyerte a szoftver-csoportot, megelőzve az örökös világbajnok Mephisto legújabb verzióját is! (Lehet, hogy az ördög elaludt?)

Ezzel olyan változás következett be a számítógépes sakkozás történetében, ami túlzás nélkül mondható forradalminak. Megszűnt ugyanis a sakkra specializált komputerek egyeduralma, a PC-programok egyenrangú ellenfelekké váltak. Hasonló frontáttörésnek lehetünk szemtanúi, mint amikor a hetvenes évek végén az óriásgépek mellé felzárkóztak a mikroprocesszoros sakkszámítógépek.

Igaz ugyan, hogy egy fecske nem csinál nyarat, sőt kettő sem, de Gideonnal és The Kinggel egyidőben más PC-programok is születtek, amelyek jóval meghaladják az eddigiek játéktudását, kapacitását. Ezt jórészt a RISC kártya felhasználásával érték el. (RISC = Reduced Instructions Software Card = csökkentett utasításkészletű szoftver kártya.) A RISC szójátéknak is beillik, hiszen „risk” angolul kockázatot jelent, és a sakkban ez igen otthonos fogalom.Rendkívüli előny, hogy a kártya szinte számítógép a számítógépben: tetszés szerinti típusú, bármilyen processzorral működtetett IBM PC-be tesszük be, teljesítménye a „házigazda” sebességétől és kapacitásától független.

A RISC kártya alkalmásában rejlő lehetőségre a sakkban már korábban rájöttek, a TASC cég kifejlesztette a „Final Chess Card”-ot (legvégső sakkártya), de annak játéktudása gyenge volt, sok hibát követett el. A Langhoz hasonló képességű és a Hegener+Glaser cég számára tevékenykedő másik kiváló programozó, a holland Ed Schröder volt az, aki a rendszer előnyeit maximálisan kihasználva és annak problémáit fokról-fokra leküzdve végül kifejlesztette a Chess Machine-nek (sakkgép) átkeresztelt Gideont. A munkát a 8 bites Mephisto programozására vállalt kötelezettsége mellett Schröder a saját szakállára végezte el. Érdemes volt.

Ugyanerre a kártyára alapozta a „The King” programját a szintén holland Johan de Koning. Lyon után ő is sikeres továbbfejlesztést hajtott végre, noha még nem rendelkezik Schröder tapasztalataival, rutinjával. Mindkét program megszületésében sok segítséget nyújtott a kitűnő holland sakk-számítógép szakértő, Jan Louwman, aki egy interjúban elmondta, hogy egyedül a Chess Machine tesztelésére mintegy 1300

munkaórát fordított! Mivel a két program használatához azonos kártya kell, most együtt hirdetik őket, hangsúlyozva, hogy jól kiegészítik egymást. A kétféle memóriájú Chess Machine közül a jobbiknak az ára meghaladja az 1000 márkát, de ha meggondoljuk, hogy úgy sakkozik, mint egy 4-5000 márkáért kapható Mephisto Lyon vagy Mephisto Vancouver, akkor ez az ár nem is olyan nagy.

Az élvonalbeli új PC-programok közé sorolható az amerikai Larry Kaufman és Don Daily alkotta Rexchess. Látván, hogy a program játékerejét önmagában nem lehet meghatározni, mert az a PC kapacitásának is függvénye, beépítettek egy „Time-Ref" (időérték) funkciót, amely kimutatja a használt számítógép sebességéhez viszonyított Élő-pontszámot. (A Élő-pontokat nemcsak a sakkozók, hanem a komputerek játékerejének szemléltetésére is alkalmazzák.) Ha a program 10 MHz sebességű gépen fut, az időértéket 1000-nek tekinti (mely esetben a program játékereje 1875 Élő-pontnak felel meg), ha lassabb a processzor, az időérték arányosan alacsonyabb, ha gyorsabb, ennek megfelelően magasabb.

A változó játékerejű programok sorában a legmagasabb szintet az ugyancsak amerikai Martin Hirsch érte el, M Chess nevű alkotásával. A program Vancouverben a szoftver csoport 2. helyezettjeként szintén a Mephisto elé szemtelenkedett. Korábbi verziói már az előző világbajnokságokon is igen jó eredményt értek el. Azoknak is érdemes megvásárolni, akiknek kisebb kapacitású PC-jük van. (A közelmúltban forgalomba került legújabb verzió csak 200 német márka.)

Befejezésül megemlíthető még a holland Frans Morsch által, a ChessBase sakkadatbank vállalattal karöltve készített (és mindössze 99 márkáért árusított) Fritz sakkprogram, amely különleges szolgáltatásaival tűnik ki: oktatásra alkalmas, ugyanakkor villámjátszmákban mestererejű. **Lindner László**

A sakk és a go

A számítógépi programok néhány egyszerű játékhoz tökéletesen alkalmasak. Néhány komplex játékot (például a backgammont) szintén a legmagasabb színvonalon játsszák. A legjobb sakkprogramok is sok élvonalbeli versenyzőt legyőznek, de Kaszparov trónját még nem veszélyeztetik. Hogyan működnek ezek a sakkprogramok? Vajon lehet-e ugyanezt a technikát a go-programokra is alkalmazni?

A sakkszámítógép játékstílusa egyáltalán nem hasonlít az emberére. Kivétel nélkül megtorolja a taktikai hibákat, mert számítási sebessége lehetővé teszi, hogy a program néhány lépés összes lehetséges variációját előre lássa, de még nincs megfelelő eszköztára arra, hogy magába sűrítse az ember stratégiai tapasztalatát.

A sakkprogram minden állásban sorra kiválasztja a lehetséges lépéseket, s megvizsgálja az arra adható ésszerű válaszokat. Ezután az algoritmus minden lépés-ellenlépés kombinációnál további lehetséges lépéseket vizsgál. Ez a folyamat előre meghatározott lépésszámig folytatódik, s a program végül kiválasztja a játékfa megadott szintjén általa legjobbnak ítélt variációt. A játékfa mélyebb elemzésével a kiválasztott lépéslehetőségek száma rohamosan gyarapszik. Például, ha egy hétlépéses vizsgálat a fa minden szintjén öt lépésjelöltet generál, ez azt jelenti, hogy a programnak 5^7, azaz 78 125 kombinációt kell megvizsgálnia.

A sakkprogram „ügyessége" azon múlik, hogy a betáplált sakktudás alapján mennyire tudja szelektálni a lépéseket, hogy ne kelljen az összes lehetséges lépést megvizsgálnia, csak azoknak elenyésző részét. Igaz, a játékfa így is tekintélyessé nőhet. Minél komolyabb a szelektálás, annál nagyobb lehet a számítási mélység, ugyanakkor egyre nagyobb az esély arra, hogy a szelektálás következtében a program az igazán erős lépést már a számítás elején eldobja, nem foglalkozik annak további értékelésével. A túlságosan szelektív algoritmusú program ezért a nyerésre álló játszmát is könnyen elveszítheti.

A go-ban azonos szintű elemzés eléréséhez sokkal hosszabb lépéssorozatból álló változatok elemzése szükséges. A gyengébb legális lépéseket viszonylag könnyen ki lehet szűrni, de egyetlen változat mélysége is elérheti a húsz lépést, lépésszintenként pedig átlagosan tíz lépést kell megvizsgálni, ami bizony összesen 1020 lépés. A jelenleg elérhető legnagyobb teljesítményű szuperszámítógép ennyi lépést legjobb esetben is csak 3 millió év alatt értékelhet ki. A go-programban ezért a „lépésjelöltek" számát drasztikusan korlátozni kell, méghozzá annyira, hogy az előretekintést — a játékfa mélyítését — fel sem lehet hatásoson használni.

A gyakorlatban a go-programok többsége nem is a játékfa felépítésére támaszkodik, hanem mintákat használ helyette. A program a tábla kis részét letapogatja, és összehasonlítja azt a memóriában tárolt adatokkal, hogy annak alapján a legjobb lépést kiválaszthassa. Az emberek ezért könnyen legyőzik az alakzatorientált programot, ha szokatlan, egyébként abnormális lépést húznak, ugyanis a memóriában lévő könyvtár kapacitása korlátozott, nem tud minden lépést tárolni. Feltételezhető, hogy egy kifinomult nagy adattár segítségével a program jól játszana, de ilyen sok adatot befogadó és gyors elérésű adattároló kifejlesztése óriási feladat. A fejlődés útja valószínűleg a feladat automatizálása, például a könyvtárát állandóan és önállóan bővítő program kidolgozása.

Kovács P. Attila

Hosszabb távon verhetetlen snóbli

Fej vagy írás?

A snóbli játék „fej vagy írás" változatának a szabályai nagyon egyszerűek: mindkét játékos takarva leteszi a nála levő érmét az asztalra, majd egyszerre felfedik tippjüket. Ha a tippek azonosak (mindkettő fej, vagy mindkettő írás), akkor az egyik, ha különbözőek, akkor a másik játékos nyer.

Könnyű belátni, hogy amennyiben mindkét játékos teljesen véletlenszerűen választ, akkor hosszú távon a játék döntetlenhez vezet. Nincs viszont olyan ember, aki erre képes lenne, vagy aki ne próbálná ellenfele játékában a szabályszerűséget felfedezni és kihasználni. A számítógépeken sem tudnak igazi véletlenszámokat előállítani: álvéletlen számokat használnak. (A kiindulási értéket általában a rendszeróra alapján határozzák meg, majd minden újabb értéket az előző alapján képeznek egy célszerűen kiválasztott algoritmussal.)

Szükségszerű, hogy egy idő után az értékek ismétlődjenek. Az ismétlődés periódusa általában jóval több mint tízezer. Játékprogramok esetén ez a legtöbb esetben elegendő, de komoly természettudományos számításoknál szükséges lehet valódi véletlen eseményből kiinduló speciális berendezésekre. Mivel az ember nagyon rosszul tudja „generálni" a véletlenszámokat, a „fej vagy írás" snóbliban már hat-nyolc tipp alapján jó becslés készíthető a következő tippről. Ezt végzi el a mágneslemez mellékleten közreadott SNOBLI program is — és így lesz legyőzhetetlen.

Pofonegyszerű

Az Alaplap elődjében, a Mikroszámítógép Magazinban három évvel ezelőtt már megjelent egy cikk a snóbliról. Akkor még a Basic-ben ZX-Spectrumra megírt program nem rajzolt, és valamivel rosszabbul is játszott, mint a mostani. És természetesen csak azok tudták használni, akik a programlista alapján fáradságos munkával begépelték az egészet. A mostani program már rajzol, képes fájlból is működni, és tippjeiről nyilvántartást vezet. Használata igen egyszerű:

SNOBLI [input_fájl] [output_fájl]

Ha a programot paraméter nélkül indítjuk el, akkor interaktív módon használható az F9 és az F10 billentyűk segítségével. Speciális szolgáltatás a statisztika készítése. Ezzel belenézhetünk a program belsejébe. Az oszlopgrafikonon egy piros és egy zöld oszlop látható. Ezek aránya mutatja, hogy a program szerint milyen valószínűséggel fog a játékos fejet illetve írást tippelni. Természetesen a gép ugyanilyen valószínűséggel tippel az ellenkezőjére. Az oszlopgrafikon többi oszlopa a tippelés egyenletességét mutatja. Ha a tippelés teljesen véletlenszerű, akkor az oszlopok közel azonos magasságúak.

A programnak meg lehet adni egy input és egy output fájlt. Az output_fájl paramétert megadva a program a tippjének megfelelően „F" (fej) és „I" (írás) betűt ír az output fájlba. Hasonló formátumú fájlt vár input_fájlként, de mivel csak minden bájt legalsó bitjeit vizsgálja, bármilyen input fájllal működőképes. A program a billentyűzetet ilyenkor is figyeli, és így lehetőség van statisztika készítésére vagy a játék folytatására.

Csalás?

Lesznek esetleg olyanok, akik a géppel játszva (és mindig veszítve) csalással gyanúsítják a programot vagy annak íróját. Fölösleges bizonygatnom, hogy egyikünk sem csal, és erről a programlista alapján könnyen meg lehet győződni. A program a saját tippjét előbb írja be az output fájlba, mielőtt a játékos csak tippelne. Tippjét a gép utólag megváltoztatni már nem tudja, különösen akkor nem, ha fájlnévként a nyomtató vagy a soros vonal nevét adjuk meg neki. A soros vonal másik végén terminállal vagy egy másik programmal lehet a játék tisztaságát ellenőrizni.

Bonyolultabb a nyomtatót használni, mert a SNOBLI NUL PRN parancssorral elindított program ugyan minden tippjét elküldi a nyomtatónak, a nyomtatók legtöbbje azonban ezt nem írja ki azonnal, hanem előbb összegyűjt egy sornyi karaktert. A még nem teljes sor kinyomtatásához a nyomtatót off-line-ba kell tenni, egy soremelést ütni, majd újra on-line-ba tenni.

A tömörített fájlok számítanak igazi ellenfélnek! A tömörítőprogramok ugyanis éppen az ismétlődő szekvenciák felismerése révén tömörítenek. Ezek akár a döntetlent is képesek elérni, ami igazán jó eredmény egy verhetetlen programmal szemben. Gondolkozzon el az olvasó: miért is ügyelnek annyira a rulettben a kerék kiegyensúlyozottságára, és miért tiltják mindenféle számítógép használatát az asztalnál?

Gép a gép ellen

Kössön össze két számítógépet soros vonalon. Programozza mindkettőt azonos sebességre és formátumra, majd indítsa el rajtuk a SNOBLI programot. (A modemjeleket vissza kell hurkolni, a szokásos keresztbe kötött null-modem kábel csak akkor működne, ha mindkét program pontosan egy időben indulna.)

A program önmagával is játszatható. Ehhez előbb hozni kell egy listát valamelyik játékról:

SNOBLI NUL LISTA.TMP

Ezután játszatni kell a gépet a létrejött fájllal:

SNOBLI LISTA.TMP

A program fegyvertára

Akik nemcsak játszani akarnak, nézzék végig a forrásprogramot is, amely négy darab C nyelvű fájlból és az ezek közti kapcsolatot biztosító header fájlokból áll.

1. A főprogram (SNOBLI.C) feladata a játék adminisztrálása, a megfelelő

rajzoló és számoló rutinok meghívása. Itt történik a billentyűzet, valamint az input és az output fájl kezelése is.

2. A kirajzolást a PIC.C végzi. A program támogatja az EGA, VGA, CGA, MCGA, EGAMONO, EGA64, HERCMONO monitorokat, ezért minden rajzoló műveletnek számos paramétere van, hogy a színek és a méretek mindegyik megjelenítőn jó összhatást keltsenek.

3. Mindenféle szövegeket találunk az ERROR.C fájlban. Lehetőséget ad angol nyelvű program létrehozására is.

4. Az igazán érdekes részt a COUNT.C fájlban találjuk, pontosabban az erről leválasztott PARAM.H fájlban. Itt lehet beállítani a játék stratégiáját.

— A SBITS mondja meg, hogy a játékos tippjét mennyire befolyásolják saját előző tippjei, az MBITS pedig azt, hogy a játékos tippjét mennyire befolyásolja a gép tippje. Nagyobb érték megadása lassítja a tanulási folyamatot, de jobb eredményhez vezet.

— QUICK paraméter a játék kezdetén segít. Ha túl kicsire választjuk, akkor a játék elején a program sokat veszít, amíg meg nem ismeri az ellenfél játékát.

— A felejtésnek különösen nagy szerepe van, ha másik játékos más stratégia szerint folytatja a játékot. Ezt szabályozza a FORGET paraméter. A listában szereplő értékek használata kb. 500-1000 játék esetén már biztosítja a program fölényét egy átlagos játékossal szemben.

— Az inicializálást initcount( ) eljárás végzi. Ez a becslő táblázatot feltölti QUICK értékkel, azt szimulálva, mintha a programot már régen elindították volna és mindeddig egyenletesen játszottak volna ellene.

— A tippelést a guess( ) eljárás végzi. Ez a táblázatból megállapítja, hogy hasonló helyzetben a játékos eddig milyen gyakorisággal tippelt fejt vagy írást, és ellenkezően tippel. Hasonló helyzetnek az számít, ha a játékos megelőző SBITS számú tippje és a gép MBITS számú tippje megegyezik.

— A táblázat folyamatos nyilvántartását az info( ) végzi. A felejtés is itt kapott helyet. A táblázat kiugró értékekeit az eljárás nagyobb mértékben csökkenti.

Pintér Gábor

Labirintus

Rejtvényfüzetekben gyakran találkozhatunk labirintusokkal. Aki szereti ezeket megfejteni, vagy éppen ilyenek készítéséből él, bizonyára örömmel fogadja a SolarSoft Programkönyvtár magyar szekciójának legújabb darabját: az M47-es lemezen két labirintusrajzoló program kapott helyet.

A labirintuskészítéshez szükséges első program a Mikroszámítógép Magazinban valamikor régen megjelent labirintusrajzoló program átdolgozása, amivel a képernyőre rajzolhatunk labirintusokat. Az új program már C-ben íródott és strukturált felépítésénél fogva könnyen adaptálható más megjelenítőkhöz is. A másik programmal labirintusokat rajzolhatunk HP7475-kompatibilis plotterre.

Persze nem sokaknak van otthon plotterük, igazi haszna a programnak nem is ez. Az újabban megjelent rajzolóprogramok és kiadványszerkesztők a HPGL formátumú fájlokat tudják olvasni, ezáltal az elkészített labirintust beépíthetjük akár egy újságba is. (Különösen alkalmas erre a Corel Draw 2.0.) A beolvasott HPGL formátumú labirintust átszerkeszthetjük, a Windows által támogatott nagyszámú nyomtató valamelyikén kinyomtathatjuk, vagy éppen más formátummá átkonvertálhatjuk.

Milyen is egy „jó" vagy „szabályos" labirintus?

1. Egy bejárata és egy kijárata van.

2. A bejárattól a kijáratig egyetlen út vezet.

3. A bejárattól a labirintus bármely pontjára el lehet jutni.

A program a következő algoritmust használja. Induljunk ki a még felosztatlan táblából. Véletlenszerűen választott helyen osszuk ketté a téglalapot egy olyan fallal, amelyen pontosan egy átjáró van. Ismételjük meg ezt az eljárást a jobb és a baloldali téglalapra, majd végezzük el ugyanezt a keletkezett négy téglalapra. Folytassuk az eljárást mindaddig, míg a téglalapok tovább már értelmesen nem oszthatók. Az így kapott labirintus a kívánalmaknak minden szempontból megfelel.

Más úton is megközelíthetjük a problémát. Ha megvizsgálunk egy szabályos labirintust, láthatjuk, hogy az két összefüggő fal-rendszerből (elágazási fából) áll. A kettő között halad a labirintuson átvezető út. Ha tudunk írni olyan programot, amely ezt a két véletlenszerű fát megrajzolja, akkor lényegében már meg is oldottuk a feladatot. Ilyen fát a következőképpen rajzolhatunk. Jegyezzük fel egy naplóba a fa meglévő ágainak azon pontjait, ahonnét további ágakat tehet növeszteni. Minden lépésben válasszunk ki véletlenszerűen egy-egy pontot. Vizsgáljuk meg, hogy a négy lehetséges irány közül továbbléphetünk-e valamelyikbe. Ha nem, töröljük a pontot a naplóból — egyébként pedig az egyik lehetséges irányba növesszük tovább az ágat, és a kapott új pontot jegyezzük be a naplóba. Amikor elfogynak a folytatható ágak, a labirintus rajzolása befejeződött.

Ezzel a módszerrel a programozás egy kicsivel bonyolultabb. A program ezért az előbbi algoritmust használja. Fordítását Borland C++ 2.0-val végezhetjük. A program fő részét a LAB.C fájban találjuk. Ennek magját az xfelez( ) és az yfelez( ) eljárások alkotják. Ezek osztják ketté a téglalapot vízszintes ill. függőleges irányban. A két szubrutin olyannyira hasonló, hogy működésük egységesen tárgyalható.

Az eljárás paraméterként megkapja a kettéosztandó téglalap adatait: a bal felső sarok koordinátáit, a téglalap szélességét és hosszúságát.

A tényleges rajzolást a külön fáljban (IF.C) megírt interfész rész végzi. Ez a szétválasztás teszi lehetővé, hogy a képernyőre és a HPGL fájlba rajzoló program fő része megegyezzen. Az interfész részben megtaláljuk a rajzoláshoz szükséges minimális számú rutint. Ezek egyszerű feladatokat látnak el, így könnyen átírhatók más megjelenítőre is.

Mindkét program használatakor lehetőség van a maximális méretek, a folyosószélesség és a bonyolultság beállítására a DOS SET parancsaival.

Pintér Gábor

Egy kis „daktiloszkópia”

Az IBM PC-t „boncolgató" sorozat elkövetkező részeiben végiglépkedünk a legelterjedtebb perifériákon, valamint az alapkártya programozható elemein. A témakörök sorrendje néhol logikátlannak tűnhet majd, de a tárgyalási sorrend meghatározásakor a minél jobb érthetőség az első szempont. Tehát most a második rész következik, a jelszó pedig: klaviatúra. Azok az ábrák és táblázatok, amelyekre a szövegben több helyen is utalunk, terjedelmi okokból a mágneslemezen kaptak helyet. Az írás áttanulmányozása előtt ajánlatos ezeket kinyomtatni.

Az IBM PC billentyűzete önmagában is intelligens egység. Bonyolultságára jellemző, hogy saját mikroprocesszort tartalmaz. Belső felépítésével itt most nem foglalkozunk, csak azokkal a jellemzőkkel, amelyek szoftverúton megváltoztathatóak.

Az XT és az AT típusú gépek billentyűzetének működése jelentős eltérést mutat. A továbbiakban ahol nem jelezzük külön, hogy az információ az XT vagy az AT típusra vonatkozik, ott a két rendszer nem különbözik egymástól.

A billentyűzet gombjainak egybájtos sorszámuk van. Minden billentyűsorszám egyedi, és nem függ a billentyű burkolatára rajzolt jeltől. A 84, 101 és 102 gombos klaviatúrák billentyűinek sorszámozását az 1. ábra mutatja.

Amikor lenyomunk egy billentyűt, akkor a belső vezérlőegység ezt érzékeli, majd megállapítja a sorszámát. Minden billentyűsorszámhoz tartozik egy ún. scankód. A klaviatúra előállítja a lenyomott billentyű scankódját, és ezt küldi el a számítógép felé.

A billentyűsorszám és a scankód összerendelését a két rendszer más-más módon valósítja meg. A konkrét konverziós értékeket a 2. táblázat mutatja.

Az XT „bedrótozva" rendeli össze a két kódot. Minden billentyűsorszámhoz mindig ugyanazt a scankódot szolgáltatja. A billentyű lenyomásakor a klaviatúra ezt a scankódot küldi a számítógép felé. Ha a billentyűt lenyomva tartjuk, akkor a scankód bizonyos időközönként újra meg újra kiküldésre kerül. A billentyű felengedését a billentyűzet a scankód újraküldésével jelzi a gépnek. A scankódok az átküldött adatbájt alsó 7 bitjét foglalják el. Az átküldött bájt legfelső bitje a gomb lenyomásakor és lenyomva tartásakor 0, felengedésekor pedig 1 logikai értéket tartalmaz.

Az AT billentyűzete egy kicsit bonyolultabb. A sorszámok és a scankódok összerendelésének kérdésében három lehetőség közül választhatunk.

1. Minden egyes billentyűsorszámhoz tartozik egy alap-scankód. Van néhány billentyű, amelyhez több scankód is tartozik. Ezeknél a billentyűknél a kiadott scankódot befolyásolja a Shift, Ctrl, Alt és Num lock billentyűk állapota is. Az ismétlési funkció az XT klaviatúrával megegyezően működik. (A konverziós értékek a 2. táblázat 4. oszlopában láthatók.)

2. Minden egyes billentyűsorszámhoz tartozik egy alap-scankód, de ez különbözik az előző pontnál ismertetettől. Van néhány billentyű, amelyhez több scankód is tartozik. Ezeknél a billentyűknél a kiadott scankódot befolyásolja a Shift, Ctrl, Alt és Num lock billentyűk állapota is. A billentyű lenyomásakor és lenyomva tartásakor a scankód kerül kiküldésre, felengedésekor pedig előbb a hex. F0 prefix, majd a scankód. Folyamatos nyomva tartás közben nem kerül a kód ismételt kiküldésre. Bekapcsolás után az AT gép billentyűzete ebben a módban dolgozik. (A konverziós értékek a 2. táblázat 5. oszlopában láthatók.)

3. Minden egyes billentyűsorszámhoz tartozik egy alap-scankód, de ez különbözik az 1. és a 2. pontnál ismertetettektől. Nincs olyan billentyű, amelyhez több scankód tartozna. A billentyűismétlési funkciók beállításának módjai a következők: typematic (scankód kiküldése lenyomáskor és ismételgetése felengedésig); make/break (scankód küldése lenyomáskor, scankód+128 értékű bájt küldése felengedéskor); make only (scankód küldése csak lenyomáskor); typematic/make/break (scankód kiküldése lenyomáskor és ismételgetése felengedésig, felengedéskor pedig scankód+128 értékű bájt küldése). Az ismétlési funkciókat beállíthatjuk globálisan, minden gombra vonatkozóan, valamint billentyűnként egyesével is. (A konverziós értékek a 2. táblázat 6. oszlopában láthatók.)

A klaviatúra mikroprocesszora által megállapított scankód az XT és az AT gépeknél más-más módon kerül a számítógépbe. Mindkét géptípus billentyűzete soros vonalon továbbítja a kódot.

Az XT gépekben a soros jelet egy 74LS322 vagy hasonló rendeltetésű IC alakítja párhuzamossá. Ez a bájt beíródik a 8255 Párhuzamos Periféria Interface IC 'A' jelű portjára (i/o címe hex. 60). Ezután a megszakításvezérlő, értesülve a scankód megérkezéséről, a 09 sorszámú megszakítás végrehajtását kéri a központi processzortól. A megszakítási vektor a klaviatúrát kiszolgáló BIOS-rutinra mutat (ha mi át nem állítottuk). Ennek a rutinnak kötelessége kiolvasni a scankódot a 8255 'A' portjáról, majd a kiolvasás tényét közölni a rendszerrel. Ez utóbbi művelethez először ki kell olvasni a 8255 'B' csatornáját (i/o címe hex. 61). A kapott bájt legfelső bitjét be kell kapcsolni, majd kiírni a 'B' portra. Végül vissza kell írni a 'B' port eredetileg beolvasott értékét.

Az AT gépekben a billentyűzet kezelését egy 8042 típusú processzor látja el. A billentyűzetről érkező soros jeleket fogadja, majd párhuzamossá alakítja, és programozásától függően még egy konverziót is végez rajta (lásd később). Ezt a jelet elhelyezi a saját kimeneti pufferében (i/o címe hex. 60), és a státusregiszterének (i/o címe hex. 64) legala-

csonyabb bitjét bekapcsolja. Ezután a megszakításvezérlő áramkörön keresztül kéri a központi processzortól a 09 sorszámú megszakítási rutin végrehajtását. Ez a rutin fogja leemelni a 8042 kimeneti pufferéből a scankódot. A scankód kiolvasásakor a státusregiszter alacsony bitje automatikusan nullázódik.

Eljutottunk arra a pontra, hogy a 09 sorszámú interrupt mögött lévő BIOS-rutin kiolvassa a lenyomott, lenyomva tartott vagy éppen felengedett billentyű scankódját. Ettől a pillanattól kezdve a két géptípus gyakorlatilag nem mutat különbséget. A BIOS megvizsgálja, hogy a beérkezett scankód speciális billentyűhöz tartozik-e. Ha igen, akkor végrehajtja a kívánt funkciót. Ha valamelyik Shift vagy Num lock, Scroll lock, Caps lock, Alt, Ctrl billentyű lett lenyomva, akkor ennek tényét bejegyzi az ún. Shift-státusz bájtba (lásd a BIOS-t ismertető bekezdésben). Vizsgálja ezenkívül az Alt-Ctrl-Del, Ctrl-Num lock, Ctrl-P, Ctrl-Break, Shift-PrtScr, Alt-numerikus csoport billentyűkombinációkat is. A beérkező kódból a vizsgálat elvégzése után ASCII-kódot képez a BIOS. A scankódot, valamint ezt az ASCII-kódot helyezi el aztán a DOS billentyűzetpufferében. Az ASCII-kóddal nem rendelkező billentyűknél az ASCII-kód helyére nullát ír. A billentyűzetpuffer 32 bájt hosszú, de ebből 2 bájt vezérlőinformáció. Mivel egy billentyű 2 bájt információt jelent a pufferben, összesen 15 billentyűleütés tárolható. Ennyivel előzheti meg a klaviatúrainput a fizikai igényt. Ha a puffer betelt, a BIOS akkor is kiolvassa a leütött gombok kódjait a PPI-ből, illetve a 8042-ből, elvégzi a különleges kombinációkra vonatkozó ellenőrzést, de a ilyenkor a hangszórót szólaltatja meg. Ekkor a billentyűleütés elvész, hacsak nem olyan speciális kombináció volt, amelyet nem is kell a pufferbe helyezni.

A klaviatúrát kezelő különböző DOS-funkciók és a magas szintű programozási nyelvek ide vonatkozó utasításai mind a billentyűzet pufferével tartják a kapcsolatot. Innen olvassák be a karaktereket, és ide írják ki a szimulált billentyűket is.

A BIOS klaviatúrakezelő funkciói a hex. 16 sorszámú megszakításon keresztül érhetőek el. A megszakítás három dokumentált funkcióval rendelkezik. A megszakítás aktivizálásakor a végrehajtatni kívánt funkció kódját az AH regiszterben kell elhelyezni. A 0 sorszámú funkció kiolvas egy karaktert a billentyűzetpufferből, és annak scankódját az AH, ASCII-kódját pedig az AL regiszterben helyezi el. A kiolvasott kódokat a pufferből törli. Ha a puffer üres, a funkció megvárja, hogy lenyomjunk egy billentyűt, és csak ezután tér vissza. Az egyes sorszámú funkció két dologban különbözik az előzőtől. Egyrészt ha a puffert üresen találja, akkor a zéró bitet bekapcsolja, és azonnal visszatér. Ha van kiolvasandó karakter, akkor kiolvassa, de nem törli azt ki a pufferből. A kiolvasott karakter kódjait ugyanúgy adja vissza, mint a nullás funkció. A kettes sorszámú funkció a billentyűzet státusát kérdezi le. Visszatérési értéket az AL regiszterben szolgáltat. A regiszter nyolc bitje nyolc billentyűt reprezentál. A bekapcsolt biteknek megfelelő billentyűk aktívak. A billentyű-hozzárendelések a következőek (bit sorszáma — jelentése): 0 — jobb shift lenyomva, 1 — bal shift lenyomva, 2 — Ctrl lenyomva, 3 — Alt lenyomva, 4 — Scroll lock aktív, 5 — Num lock aktív, 6 — Caps lock aktív, 7 — Ins mód aktív.

Az XT típusú klaviatúra tárgyalását itt befejezettnek is tekinthetjük, az ezután közölt információk ezekre már nem érvényesek. Az AT gépek billentyűzete azonban programozható jellemzőkkel is rendelkezik. Egyrészt a kiszolgáló 8042-es processzor is felismer néhány utasítást, másrészt pedig maga a klavitúra (illetve a benne elhelyezett processzor) is programozható.

Ismerkedjünk meg először a 8042 típusjelű áramkörrel, amely önmagában is intelligens egységnek tekinthető. Ez az IC 2 kbájt ROM és 128 bájt RAM memóriát foglal magában. A rendszer adatbuszával a státusregiszteren, az input pufferen és az output pufferen keresztül tartja a kapcsolatot. Rendelkezik ezenkívül egy kimeneti és egy bemeneti porttal, valamint két tesztbemenettel.

A státusregiszter a hex. 64 i/o porton keresztül érhető el. Tartalma csak olvasható. Bitjeinek jelentését a 3. táblázat tartalmazza.

Az output puffer a hex. 60 i/o címen érhető el. Tartalma csak olvasható. A státusregiszter nulladik bitjének 1 állapota jelzi, ha érvényes adatot tartalmaz. Az adat kiolvasása után ez a jelzőbit automatikusan nullázódik.

Az input puffer csak írható. Ezt a puffert többféleképpen is elérhetjük. A hex. 64 i/o címre történő kiírás ebbe a regiszterbe kerül. Az ily módon kiírt bájtot a processzor vezérlőutasításként fogja értelmezni (az utasítások részletes listáját a 4. táblázat tartalmazza). Ha az így kiírt utasításhoz adatbájt is tartozik, akkor azt a hex. 60 i/o címen keresztül kell kiküldeni. Ha viszont a hex. 60 címre olyan adatot írunk, amely nem egy előzőleg a hex. 64 portra kiadott utasításhoz tartozik, akkor ez a bájt mint vezérlőparancs továbbkerül a billentyűzetegységhez (lásd később).

A processzor bemeneti portjára kapcsolódik többek között az alapkártyán lévő, a monitor típusát kiválasztó kapcsoló, az alapkártyán lévő RAM-terület méretét mutató jumper, valamint a doboz előlapján elhelyezett billentyűzetet letiltó kulcs kapcsolója. Ez a port nincs közvetlenül bekötve a központi processzor i/o címterébe. Hogy tartalma mégis olvasható, azt a 4. táblázat mutatja. A kiolvasott értékek értelmezését a 5. táblázat magyarázza.

A kimeneti port két vonala tartja a soros vonali kapcsolatot a billentyűzettel, egy másik vonal szolgál a kimeneti pufferben lévő adat jelzésére (ennek a vonalnak az aktív szintje váltja ki a 09 interruptot). A vonalak részletes felsorolását a 6. táblázat tartalmazza. Ez a port nincs közvetlenül bekötve a központi processzor i/o címterébe.

Mint már szó volt róla, a 8042 IC input pufferén keresztül lehetséges a billentyűzetegység részére vezérlőkódokat küldeni. E vezérlőkódoknak a részletes felsorolása a 7. táblázatban olvasható. A táblázatban külön jelölve vannak azok a parancsok, amelyeket csak 101/102 gombos klaviatúrák ismernek fel.

Előfordul, hogy a billentyűzet akkor is küld adatot a rendszernek, amikor nincs lenyomva billentyű. Ezek olyan vezérlőinformációk, amelyeket a billentyűzet küld a rendszer felé. Kevés kivételtől eltekintve nem kerülnek be a 8042 kimeneti pufferébe, hanem a vezérlőegység a saját hatáskörében intézkedik a parancs nyugtázásáról és végrehajtásáról. A 8. táblázat ezeknek a kódoknak a felsorolását is tartalmazza.

Fridl György

NETREND RT

1089 Budapest, Elnök u. 1.
Tel: 113-8217; 133-4760 • Fax: 113-9537

Alapkonfigurációk

- 1 megabájt RAM
- Baby AT-ház
- 200 W-os tápegység
- 101 gombos billentyűzet
- 40 Mbájtos HDD
- 1,2 Mbájtos FDD
- IDE kontroller
- Soros, párhuzamos csatoló
- Monomonitor és csatoló
- Alaplap

– AT 286-12/16	54 500
– AT 286-12/16 EMS	55 800
– AT 286-16/21 EMS	58 000
NEAT 286-20/26 számítógép	61 050
– 386-40/65, cache	87 500
– AT 486-33/150 ISA	
– 256 kbájt cache	209 000
– AT 386-20/25	74 500
– AT 386-25/33	77 500
– AT 386-33/58	84 900
– AT 486-25/117	
– 128 kbájt cache	119 500
– 486-33/150 EISA	
– 256 kbájt cache	409 000

Hálózati terminál:

NEAT 286-12/16	**48 750**
NEAT 286-16/21	**56 500**

Monitor-csatolókártyák:

Monokróm	1 400
CGA	1 800
EGA	4 900
VGA 800x600	7 900
VGA 1024x768	10 500
VGA 1024x768	11 600
VGA 1024x768	14 900

Monitorok:

Egyszínű (borostyánsárga)	8 700
Egyszínű (papírfehér)	9 500
EGA	27 000
VGA (1024x768)	31 500
VGA egyszínű, 1024x768	18 500
VGA Multisync	44 000

MFM, ESDI és IDE winchesterek nagy választékban!

Házak:

Baby AT + 200 W táp	6 200
LED-es baby AT + táp	8 100
Slimline + táp	8 100
Minitorony + táp	9 300

Szünetmentes áramforrások:

UPS 400 VA NOVELL	38 800
UPS 550 VA	26 500
UPS 600 VA NOVELL	44 800
UPS 1 kVA	42 900
UPS 1,2 kVA NOVELL	98 500
UPS mon. kártya	7 500

Nyomtatók:

FX-850	49 500
FX-1050	48 750
LQ-850	76 400
LQ-2500+	129 000
DFX-500	183 500
DL 5600	195 000
HP LASERJET III.	189 000
FX-1000	39 000
FR-15	44 200
LC-200 színes	33 500

RAM-ok:

4164-10	140
41464-08	290
41256-08	145
41256-06	280
44256-08	460
SIM, 1 Mbájt	3 930
SIP, 1 Mbájt	4 400
41256-07	199
SIM 256	1 150
SIP 256	1 490

Koprocesszorok:

80287-10	9 900
80287-12	18 000
80287-20	19 900
80387-25	39 800
80387-33	49 900

Modemek:

2400 baud belső	8 900
2400 baud külső	14 500
2400 baud MNP-5	15 000

Telefax/modem:

9600/2400 baud	22 000
8 felhasználós hálózati telefax	72 000

Egerek, scannerek:

GM-6000 egér	4 500
Series 9	4 600
Logitech soros	4 990
Logitech scanner	19 900
Catchword karakter-felismerő program	18 900
HP SCANJET PLUS	259 000

Plotterek:

Teljes HP termékcsalád

ARCnet-kártyák:

8 bit ZOT	4 300
16 bit LIN DATA	8 200
16 bit ZOT	8 200

ETHERNET-kártyák és tartozékok:

8 bit NE-1000	8 950
8 bit DE-100	9 000
8 bit DE-150	13 500
16 bit NE-2000	10 500
16 bit DE-200	14 500

ARCnet-kiegészítők:

Passzív HUB	990
Aktív HUB (int 4)	5 800
Aktív HUB (ext)	10 900
Aktív HUB + kártya	6 900
Csatlakozókábel	1 000
BOOT-EPROM	1 500

Ethernet-kiegészítők:

Transceiver	21 500
Transceiver BNC	21 000
Repeater (2 port)	41 900
BNC dugók, csatlakozók, aljzatok, fali csatlakozók	

Hálózati szoftverek:

Novell NetWare	
V.2.2/5	72 000
V.2.2/10	161 000
V.2.2/100	444 000
V.3.11/20	282 000
V.3.11/100	565 000
V.3.11/250	1 010 000
NACS	110 000
Asyncron Remote Bridge Program	29 600
D-Link Lansmart V. 3.0 op. rendszer	39 900
Elect. Mail	11 000
D-Link Bridge	29 600
ACS D-Linkhez	28 000
Remote Access	19 600
Screen monitor	15 000

LAPTOP-ok:

LT-3400 (NEAT) 40 MB HDD	178 000
CP-8100V (386) 100 MBB HDD	275 000

Szoftver

MS-DOS 5.0	4 500
Windows 3.0	5 500
MS-DOS 5.0 + Windows 3.0	9 500
Magyar paraméterezhető karakterkészlet MS Word/HP Laserjethez	16 000
50 és 93 ohmos koaxkábelek	90 Ft/m

És még sok minden egyéb...
Áraink csak készpénzfizetésre, telephelyünkön történő átvételre vonatkoznak!
A Netrend Rt. a Novell Inc., a CADKEY, valamint a Santa Cruz Operation hivatalos dealere.
Vállalkozunk komplett hálózati rendszerek szállítására, igény szerinti kiépítésben.
Komplex rendszerfelügyelet (hálózati is), szaktanácsadás, hardver- és szoftverkarbantartás.
CAD, DTP rendszerek kiépítése, szükség esetén üzemeltetése. Kérje részletes tájékoztatónkat!
Speciális Etherneteszközök, fénykábelek, fénykábeles kiegészítők EISA Ethernet és diszkkontrollerek, különböző méretű cache RAM-mal. Irodabútorok, fénymásolók, telefaxok nagy választékban
Egyedi és eseti javítások, hálózati installálások Pocket és UTP-hálózati csatolók igény szerinti szállítással.
Áraink az ÁFA-t nem, de a 12 hónap csereszavatosságot tartalmazzák.
Kedvező lízingfeltételek!

INFORMÁCIÓKÉRÉS: 12 ▲

Hozzáférés — két akadályon áthágva II.

A szűrő első lépcsője

Az előző hónapban indult sorozat — a bevezetése keretében — akkor az alapfogalmaknak tisztázásáig jutott. Most már a tárgykör belsejébe merészkedhetünk; lássuk, miként épül fel és működik a szűrő első lépcsője!

A szűrő feladatairól általában

Nem csupán azért neveztük első lépcsőnek, hogy valahogyan megkülönböztessük a második lépcsőtől, hanem ezzel is ki akarjuk fejezni, hogy ez a szűrő van legközelebb a gép lelkének nevezett tartományokhoz, tehát itt azokat a kéréseket szelektáljuk, amelyek közvetlenül a BIOS-hoz fordulnak mindenféle ügyekben. Ez a szűrés a hozzáférést hivatott megfékezni, de ezt könnyedén megkerülhetjük, ha saját rendszerlemezünket betéve az A meghajtóba, rendszert hívunk. (A teljesen zárt védelem tehát csak egy kevéske hardvermódosítással, illetve -kiegészítéssel oldható meg.)

A számítógép bekapcsolása után a BIOS ROM-jába égetett rutin feladata az operációs rendszer betöltése. Ez úgy történik, hogy először megnézi, hogy van-e az A meghajtóban lemez, majd ha sikeres ez az akció, akkor az első szektort betölti. Az első szektor nem más, mint a lemezen lévő fizikailag első szektor, pontosabban a 0. sáv 0. oldalán az 1. szektor. Ezt olvassa be, majd az ott található adatokat programnak értelmezi, természetesen gépi kódúnak, és minden további nélkül végrehajtja.

A lemez e része egyébként a vírusok kedvenc táptalaja, ahol úgy tesznek, mintha ténylegesen egy operációs rendszert töltenének be, gyakran persze ezt meg is teszik, de közben bűnös dolgaikat végzik: átírnak néhány megszakítást, állandóan fürkészve ezzel a gépnél lévő gyanútlan felhasználó minden mozdulatát, és ötletes fogásokat alkalmaznak, úgymint a FAT felülírása vagy egyéb szemkápráztató mulatságok. Ha viszont nem tettünk az A meghajtóba lemezt, akkor tovább nézi a rutin a számára elérhető perifériákat, hátha akad egy valamirevaló betöltőszektor. Következik tehát a merevlemezegység vizsgálata, majd pedig, teljes kudarc esetén, a ROM BASIC indítása.

A tanulság: ha gépünkön hozzáférés-védelmet akarunk csinálni, akkor figyelembe kell venni, hogy az A meghajtóra hamarabb kinéz a gép, mint a merevlemezre, ahol pedig a hozzáférés-védelmet biztosító rutinok helyezkednek el. Az 1. lépcső további feladata a hozzáférés-védelem megvalósításán kívül tehát az A meghajtóról való rendszerindítás megakadályozása, illetve az információ megfejtésének a megnehezítése. Erre valók a következő praktikák.

A meghajtó átnevezése

XT gépeken, ha egy lemezegységünk van, nem kell mást csinálni, mint óvatosan leemelni a gép fedőlapját, majd az A meghajtóhoz vezető zsinórköteget kiemeljük a helyéről, és a B meghajtó helyére illesztjük. Ezzel azt érjük el, hogy a gép nem érzékeli az A meghajtót, hiszen neki csak a B-ről szóltak, arról pedig nem illik rendszert betölteni. Ilyenkor a merevlemezről töltődik be az operációs rendszer. Ezek után a B meghajtót átnevezzük A-ra, és rövid mosolygással nyugtázzuk az esetet.

AT gépeknél a CMOS 64 bájtjában van az az információ tárolva, hogy milyen lemezegységet kapcsoltunk hozzá. Itt tehát ezeket az adatokat kell átírni, hardvermódosítás nélkül.

A ROM módosítása

Az egyik lehetőség a BIOS-t tartalmazó ROM memóriának a kiolvasása — és mivel sokféle típus van forgalomban, biztos, hogy egy eddig még ismeretlent sikerül kifognunk; tehát kezdhetjük verejtékesen visszafejteni, majd a megfelelő helyen átírni a programocskát, és beégetni egy újabb ROM-ba. A módosítás lehet például egy jelszólekérés, amely a billentyűzetről vár néhány karaktert, hogy összehasonlítsa egy szintén beégetett, tehát nehezen változtatható kulcsszóval. Ha egyezik, akkor az eredeti BIOS-ra adja a vezérlést, mely most már teljes nyugalommal hozzálát az operációs rendszer betöltéséhez, kezdve a sort az A meghajtóban esetleg elhelyezkedő lemez vizsgálatával. A kulcsszó meg nem egyezésénél fellépő megtorló intézkedésekre itt most nem térünk ki, hiszen ilyenek kiötlésének képességével minden programozó fel van vértezve.

Saját fejlesztésű kártya

A másik megoldás, hogy egy kis kártyát csináltatunk egy tanult szakember közreműködésével. Nem feledve a pókerezés kellemes perceit, valamint azt az örömöt, amikor kártyapartnerünk a feketepétert választja, itt nem a hagyományos értelemben vett kártyáról tettünk említést, hanem egy áramköri lapkáról, melynek talán a szögletes kinézete és a vele való játszadozás örömei indíthatták arra a programozókat, hogy ezt az elemet így emlegessék... Erre a lapocskára egy kis ROM-ot vagy EPROM-ot helyezhetünk, amelyet az I/O kivezetések egy szabad foglalatába helyezhetünk, ahogy például egy monitor- vagy lemezegység-vezérlőt.

Ezek a ROM, illetve bizonyos esetekben RAM memóriák tökéletesen illeszkednek gépünk buszrendszerébe, tehát olvashatjuk őket, és néha írhatjuk is, ha éppen RAM-ról van szó. Azért jó ez nekünk, mert ha a ROM úgy van befűzve, hogy azt a BIOS is érzékeli, akkor ráadódik a vezérlés, tehát a beégetett kis rutinunk elindul, hogy kedvére ténykedjék. A végén pedig újra a BIOS-nak adjuk át a szerepet. Az itt elinduló rutin nem csinál mást, mint a rendszerünk betöltődéséig nem hagyja szóhoz jutni az A meghajtót. A gépen van tehát egy olyan címtartomány, mely a képernyőmemóriák után található, és amelyet a BIOS két kilobájtonként átpásztáz: keres egy oly nagyon várt jelzést, hogy itt most egy futásra érdemes rutin van. (Hasonlóan működik az EGA-kártya is, mely — amikor a BIOS-tól megkapja a vezérlést — ki-

tölti a számára szükséges megszakításvektorokat, és egyáltalán: olyan állapotba szedi össze magát, hogy majd a gépen futó programok könnyen elérhessék, programozhassák a kedvüknek megfelelően.)

Ezek természetesen durva beavatkozásoknak tűnhetnek, de a teljes biztonságot csak ezek garantálhatják.

Hardver nélkül

Más lehetőség is adódhat. A gépünket semmiféle hardvermódosításnak és piszkálásnak nem tesszük ki, hanem a feladatot szoftveres úton akarjuk megoldani. Ilyenkor tehát lehetséges a rendszerindítás az A meghajtóról, aminek a következménye, hogy a merevlemezen tárolt adataink védtelenek a külső betolakodók rongálásával szemben. Nem tehetünk mást, mint az információt állandóan kódolva tároljuk a merevlemezünkön, és csak a memóriában fejtjük vissza. Ez tehát az egyik legfontosabb feladata az első lépcsőnek, melyet egy speciális BIOS-megszakítás átírásával, illetve magunkon történő átfűzésével oldhatunk meg igazán hatékonyan. A BIOS a saját megszakításait a memóriateszt alatt tölti ki a 0. lap megfelelő címeinek magára irányításával, tehát ezek szinte a gép bekapcsolásával azonnal élnek. Ha tehát magunkon keresztülfűzzük, akkor csak olyan kérések érkeznek a tényleges BIOS-rutinhoz, amelyeket mi engedélyezünk, vagyis átestek a szűrő 1. lépcsőjének rostáján.

Most nézzünk meg néhány BIOS-megszakítást röviden (a számok/betűk utáni kis „h" betű nem más, mint a hexadecimális szám jelölése):

10h: a képernyővel kapcsolatos dolgokat látja el;

13h: a lemezzel összefüggő I/O műveleteket hajtja végre;

14h: a soros port kezelését végzi;

16h: a billentyűzettel foglalkozik;

17h: a printerrel tartja a kapcsolatot;

1Bh: a Ctrl-Break kezelését végzi;

1Ch: felhasználói timer, mely a 08h hardvermegszakításból hívódik meg (ezzel még lesz dolgunk).

A felsorolás gyors átfutásával felismerhető, hogy a 13h-s megszakítást kell átírnunk, vagyis ide kell beláncolódnunk. Ezenkívül levédhetjük a printert is, ha a 17h megszakításba ülünk be. Ez akkor lehet okos dolog, ha gépünkhöz egy értékes lézerprinter van kapcsolva, és egyes szeleburdi felhasználóktól meg akarjuk óvni, nehogy finom mechanikája megsérüljön a gondatlan kezelés következtében.

Soroljuk fel a 13h-s megszakítás feladatait:

— az eszköz alapállapotba hozatala;

— státusinformációk lekérése;

— szektorok írása, olvasása, ellenőrzése;

— sáv formattálása, keresése.

Foglaljuk össze röviden, hogy miket kell ellátnia a szűrő 1. lépcsőjének:

a) online kódolás beláncolódva a 13h-ba;

b) a 13h-s megszakításon keresztülfutó kérések megszűrése;

c) printer levédése;

d) ha kártyánk van, akkor a floppy kiiktatása addig, amíg a merevlemezről be nem töltődik a rendszer;

e) és egy későbbre tartogatott meglepetés a vírusok ellen.

A következő részben ezeket a feladatokat és megoldásokat részletezzük.

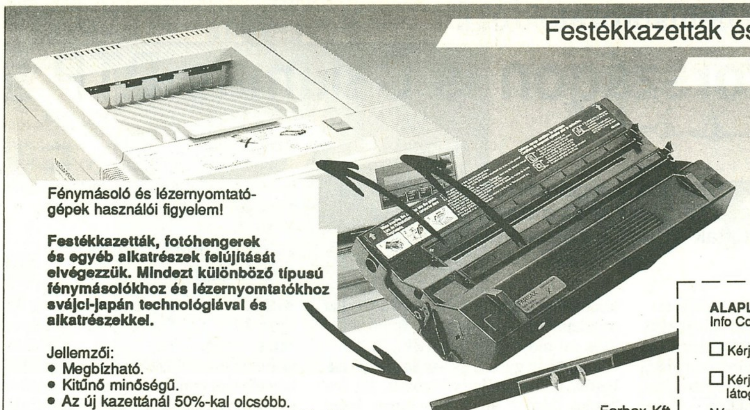
Krokovay Károly—
Radványi Tibor

INFORMÁCIÓKÉRÉS: 20 ▲

INFORMÁCIÓKÉRÉS: 26 ▲

TERMÉSZETESEN SPÓROLJON

Festékkazetták és fotóhengerek felújítása



COMPFAIR '91
A pavilon
209/5-ös stand

Fénymásoló és lézernyomtató-gépek használói figyelem!

Festékkazetták, fotóhengerek és egyéb alkatrészek felújítását elvégezzük. Mindezt különböző típusú fénymásolókhoz és lézernyomtatókhoz svájci-japán technológiával és alkatrészekkel.

Jellemzői:
- Megbízható.
- Kitűnő minőségű.
- Az új kazettánál 50%-kal olcsóbb.
- Kevesebb a hulladék, így környezetkímélő.

FARBAX®
Refill-Service

Farbax Kft.
Márkabolt és Ker. Iroda:
H-1138 Budapest, Váci út 178.
Tel.: 120-8053, Tel./Fax: 120-1241
Refill Service:
H-9700 Szombathely,
Rózsa F. krt. 2.
Tel.: 94-16839, Fax: 94-13868

ALAPLAP
Info Coupon:

☐ Kérjük részletes tájékoztatójukat

☐ Kérjük szakértőjük személyes látogatását

Név: ____________
Cégneve: ____________
Tel.: ____________
Cím: ____________

INFORMÁCIÓKÉRÉS: 21 ▶

VÁSÁRLÁTOGATÓ UTAZÁSOK

Hannoverbe, az Alaplap és a Kuoni szervezésében

CeBit '92: Utazási időpontok: március 12–14, március 14–16, március 16–18

Industrie '92: Utazási időpontok: április 2–4, április 4–6, április 6–8

Szolgáltatásaink:

— Repülőút a Lufthansa menetrendszerű járataival Budapest—Frankfurt—Budapest útvonalon
— Vasúti utazás II. osztályon Frankfurt—Hannover—Frankfurt útvonalon
— Elhelyezés kétágyas fürdőszobás szobákban (magánszállásokon), reggelivel
— Vásári belépő a kinntartózkodás időtartamára, vásári információk
— Atlasz utas- és poggyászbiztosítás az utazás időtartamára

Irányár: 51 500,–Ft/fő kétágyas szobában, egyágyas felár +2100,–Ft
Részletes tájékoztatásért szíveskedjék az alábbi kupont a Kuoni Utazási Iroda Kft. címére beküldeni.

Kuoni Utazási Iroda Kft. 1054 Budapest, V., Báthori u. 19. Tel.:132-4116, Fax:153-4350

Kérem, hogy az 1992-es hannoveri CeBit és Industrie vásárokról küldjenek részemre részletes tájékoztatást.

Cég: .... Cím: ....

Név: .... Tel: .... Fax: ....

INFORMÁCIÓKÉRÉS: 22 ▶

Sorozatunk a betűsorozatokról I.

Magyarországon is Calamus DTP

Az Atari cég ST és TT sorozatú gépeinek a forgalmazására 1990-ben a Novotrade Rt. kapta meg a disztribútori jogot. Külön üzletet is nyitottak e célból az Andrássy úton.

Az Atari számítógépeknek a szoftverellátottsága nagyon szegényes volt az első időkben, de csekély idő leforgása alatt megmozdult a gépezet, beindult a piac, és más cégek is elkezdtek programokat árulni ezekhez az új gépekhez.

Kiváló DTP rendszerével, a Calamus magyar változatával, a DTP System Kft. mutatkozott be először a tavaszi Ifabón, majd a Hungaroprinten és a CAMP kiállításon is. E cég kapta meg a program kizárólagos forgalmazási jogát a német DMC szoftverháztól, a program fejlesztőitől. A DTP System Kft. a megszokottól eltérően nem csupán forgalmazza a német szedőrendszert, hanem a magyar változatának kidolgozását is segítette. A magyar változat nem pusztán a külső megjelenési formát takarja, nemcsak a menürendszer „beszél" magyarul, hanem beépítették a magyar elválasztási szabályokat, aminek segítségével — ha nem is százszázalékosan jól működik —, könnyebbé vált a tördelő munkája. A magyar klaviatúrakiosztáson kívül folyamatosan készülnek az újabb és újabb betűtípusok magyar változatai is, amelyek mind a CWI, mind az MTI betűkiosztás szerint megvásárolhatók. Eddig 100 magyar fontkészlet közül választhat a vásárló, de folyamatosan bővül ezek száma, ugyanis a német változathoz körülbelül 1700 kapható már. A most induló sorozatunkban ezt a kiadványszerkesztői rendszert, és a hozzá ajánlott perifériákat, kiegészítő programokat ismertetjük.

Írónád a munkaasztalon

Bármelyik közismert számítógépen futó DTP rendszer közül a Calamus Desktop Publishing az egyik legprofesszionálisabb DTP program (lásd 1. ábra). Rendkívül sok lehetőséget és funkciót foglal magában. Ahogy réges-rég, a kolostorokban a készülő kódexek fölé görnyedő szerzetesek kincsként becsülték meg a legprímább nádtolljaikat (kalamusaikat), hiszen a jó munkaeszközt mindennél többre tartották, így örülhetünk mi most a Calamusnak…

Menüi nagyon egyszerűen használhatók, piktogramjai világosak, jól érthetőek (lásd 2. ábra). Egy lapon, keretekkel meghatározott területeken — több szinten — egyszerre dolgozhatunk szöveggel, léniával, raszterrel, vektor- és pixelgrafikával. Többféle mértékegységben (cm, pica pont, cicero pont és inch) adhatjuk meg a szükséges paramétereket, tetszés szerint keverhetjük is azokat. A szedési, tördelési utasításokat egérrel választhatjuk ki. A precíz montírozást könnyen kezelhető koordináta-rendszer segíti elő.

A különböző adatoknak (szöveg, elforgatott szöveg, raszter, grafika stb.) kijelölt területek mérete és helye, külön-külön és összekapcsolva is, bármikor kívánság szerint változtatható. A változás koordináta megadásával vagy egérrel vezérelhető. Természetesen az adatok követik az új területmeghatározást. Minden adatterületről — a benne lévő dokumentummal együtt — készíthető fizikai és virtuális másolat. Import-export funkciója sokféle fájlformátumú szöveg és grafika feldolgozását teszi lehetővé. Kisegítő grafikai és betűszerkesztő programjai megbízhatóak és természetesen tökéletes kompatibilitást biztosítanak.

Jelenleg 1700 vektorfont közül választhat a felhasználó. A dokumentumok közvetlenül nyomtathatók lézerprinteren és RIP nélkül levilágíthatók DTC Desktop Setteren 2540 dpi-ig, DMC interfésszel más típusokon is.

A Calamus monokróm változatának jelenlegi ára 32 930 Ft + ÁFA, a négyszínbontást tudó Calamus SL pedig 119 900 Ft + ÁFA-ba kerül. Külön kapható még LAN-hálózati rendszereken futó JOB (munka) verzió, amelyet tudástól függően 117 000 és 165 000 Ft + ÁFA- ért vásárolhatunk meg. Fontokat külön rendelhetünk típusonként 5-10 000 Ft + ÁFA-ért. 33, illetve 100 fontos egységcsomag vásárlása esetén a cég jelentős árkedvezményt ad.

A művészeten kívül is művészien

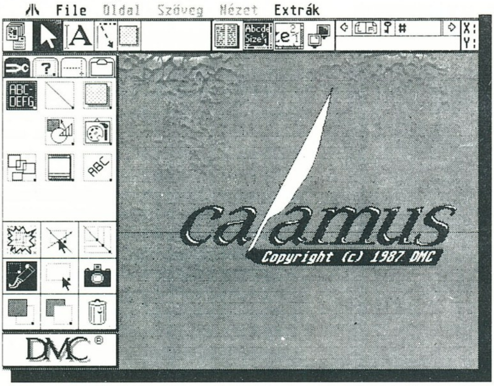
A Calamus DTP program szerves kiegészítője az Outline Art szoftver, amely vektororientált, új generációs grafikai munkaeszköz. Sokrétű alkotási lehetőséget nyújt a programban található számos különleges funkció. A Calamus-hoz hasonlóan menüvezérelt, felhasználóbarát a program.

A program komplett vektorgrafikai szerkesztő egyenesekhez, Bezier-ívekhez, kontrollutakhoz stb. Segítségével szabadon meghatározott rasztermezőket generálhatunk, grafikákat általunk meghatározott középpontból sugárirányban kivetíthetünk. A programot a betűk és szövegek tulajdonságainak, sajátosságainak és jellemzőinek korlátlan változtathatósága jellemzi. Írhatunk raszteres, átlátszó, körvonalas (kontúros), elforgatott, széthúzott, összepréselt és döntött stílusban. A kölönféle betűtípusokkal kör, ellipszis, háromszög és több más, előre megadott útvonalra írhatunk. Amennyiben egyik sem megfelelő az előre megadott alakzatok közül, akkor saját magunk új utat is meghatározhatunk. A szöveget nemcsak síkbeli útra helyezhetjük, hanem térbeli felületre is illeszthetjük (lásd a 4., 5. ábrát). A szöveget átalakíthatjuk vektorgrafikai képpé. Az Outline Art programban a Calamus összes betűtípusát használhatjuk.

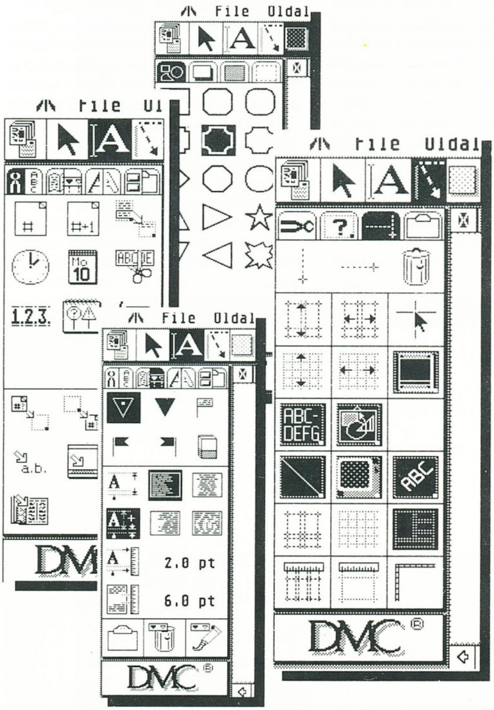
A szoftverbe programozható számológépet is építettek, amellyel a grafikák bármilyen — általunk egyenlettel megadott — két- és háromdimenziós felületre vetíthetők. A számítógépnek van egy alapkönyvtára, amely az eltolás, elforgatás, tükrözés, nagyítás, kicsinyítés, henger-, kúp- és gömbfelületre való vetítés egyenletét tartalmazza. Ha sokszor alkalmazunk más típusú leképezést, akkor lehetőség van az alapkönyvtár bővítésére is.

A Hungaroprint kiállításon a program meghirdetett ára 17 250,— Ft + ÁFA volt, s ez azóta sem változott.

Kovács P. Attila


File Oldal Szöveg Nézet Extrák
calamus
Copyright (c) 1987 DMC
DMC


Natürliches
AQUA
Mineralwasser
Enteisent und mit Kohlensäure versetzt
ATARI
PORTFOLIO
FORELLE
Symphonie No.9 (The Choral Symphony)
Ludwig von Beethoven
O-Line
O-Line
The New York Philharmonic Orchestra
O-Line
OUTLINE ART


File Oldal
File Ol
File Oldal
File Oldal
2.0 pt
6.0 pt
DMC
DMC
DMC

NAV vagy CPAV?

A két konkurens antivírus

A Norton Antivirus 1.0 tavaly télen jelent meg a piacon — már a Peter Norton Computing új szállásadójának, a Symantec cégnek a zászlaja alatt. A Central Point, a nagy sikerű PC Tools gyártója sem akart kimaradni a jól jövedelmező vírusüzletből, így idén tavasszal kijött a Central Point Anti-Virus 1.0 nevű programmal. Nortonék közben továbbléptek, és már az 1.5-ös verziót hozták ki. Ezúttal tehát ezt a két legfrissebb változatot vetjük össze ebben a rövid cikkben. Aki nem fél a vírusoktól, kérjük, azonnal lapozzon tovább az újságban, hiszen számára semmi újjal nem tudott szolgálni a két nagy segédprogram-milliomos.

Lássuk előbb a nagyobb múltra visszatekintő Norton Antivirust (a továbbiakban NAV)! Még az installálás előtt érdemes lefuttatni a NAV-ot (melyet Nortonék nemes egyszerűséggel vírusklinikának is neveznek), hogy biztosak lehessünk, nem fertőzött gépre tesszük-e fel a programot. Az eredeti rendszerlemeznek nem lehet baja, hiszen fizikailag írásvédetté van téve, vagyis nincs is kivágás a floppyn.

Ellenőrzés egy menetben

A program lényegében két részből áll: az interaktívan is használható víruskeresőből, a Virus Clinicből és a rezidensen betöltendő vírusdetektorból, a Virus Interceptből. A vírusirtás nortoni alapkoncepciója: harcolni a már ismert vírusmutációk ellen; egy program indításakor a betöltés folyamata során vírusellenőrzést hajt végre a NAV_.SYS. Egy naplózó rész az összes vírusokkal összefüggő rendellenességet megörökíti egy szöveges állományban. Novell és IBM Token-Ring hálózaton is használható, a konfigurációs állománya jelszóval is védhető. A megtalált vírusokat (amennyiben nem okoztak maradandó károsodást) eltávolítja a beteg állományokból, de ha erre már nincs mód, akkor gondoskodik a fertőző góc teljes megsemmisítéséről.

Egyszerű, menüvezérléses a program, és egérrel is dolgozhatunk. A program vírusadattára folyamatosan frissíthető — kézzel és update állományokból is. Egy vírus megtalálásakor kérhetünk olyan üzenetet is, melyet a felhasználó kíván: ilyenkor például az üzenetbe beleírhatja, hogy melyik belső telefonszámon érhető el a cég vírusirtó guruja.

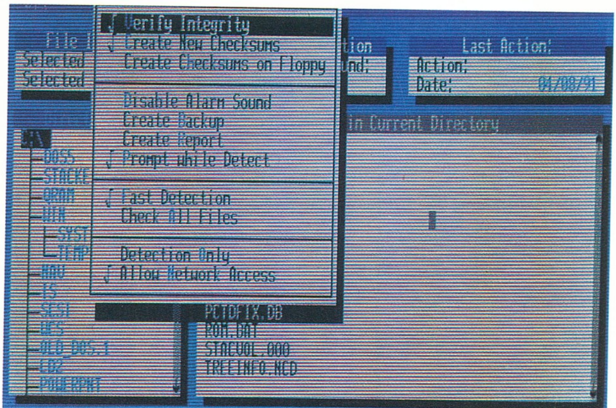
A gépen található összes meghajtó egyetlen menetben is ellenőrizhető (Scan All Drives). A fertőzött állományok egyenként és csoportosan is gyógyíthatók. Az Inoculate nevű szolgáltatás első indításkor minden egyes .SYS, .COM, .EXE és .OVL állományról készít egy ellenőrző összeget tartalmazó checksum fájlt, mellyel a legközelebbi indításkor összehasonlítja az aktuálisan számított értéket, így a program az ismeretlen vírusokkal is sikerrel veszi fel a küzdelmet. Az így készített biztonsági állományok egyetlen paranccsal eltüntethetők, egyidejűleg minden egyes alkönyvtárból.

Másfélszer annyit ér

Az 1.5-ös NAV természetesen újdonságokkal is szolgál. Ezek a következők.

Kisebb méretű a rezidens kód. Három változat közül választhatunk: eredeti méret (kb. 38K: NAV_.SYS), 4K-s (NAV&.SYS /B) és az 1K-s (NAV&.SYS) közül. A 38 K-s változat nemcsak indításkor, hanem másolás közben is ellenőriz, továbbá figyelemmel kíséri a bootszektort. A 4 K-s verzió is ügyel a bootszektorra, de már csak az elindított fájlokat vizsgálja, míg az 1K-s változat már nem vigyáz a bootszektorunkra (amelyet például előszeretettel fertőz meg a Stoned vírus). Az ismert vírusokról egy listát is kérhetünk. Nortonék állítása szerint már több mint 700 egyedi víruskódot tartanak nyilván és detektálnak!

Mód van arra is, hogy a kézi ellenőrzés során ne vizsgáljon a NAV minden állománytípust, csupán a végrehajtható



fájlokat: .COM, .EXE, .OV?, .DRV, .SYS. Az új NAV már feltölthető 640 K feletti RAM-régiókba is, és már engedélyezi más tárrezidens programok betöltését és eltávolítását is. (Ezen a téren a korábbi változatban sok probléma adódott.)

A Microsoft Windows 3.0-jával jó barátságban együtt tud dolgozni, csak az Inoculate funkciót kell hatástalanítani (erre egy apró kis BAT állomány is elég, mely a WIN indítását közrefogja, s az említett opciót ki- és bekapcsolja). Windows alatt a karakteres üzenetkiírás helyett hangjelzésekkel figyelmeztet a NAV. Egy jópofa ikont is kapunk a Windows-os felhasználás színesítéséhez (NAV.PIF, NAV.ICO).

A „középpont" célpontjai

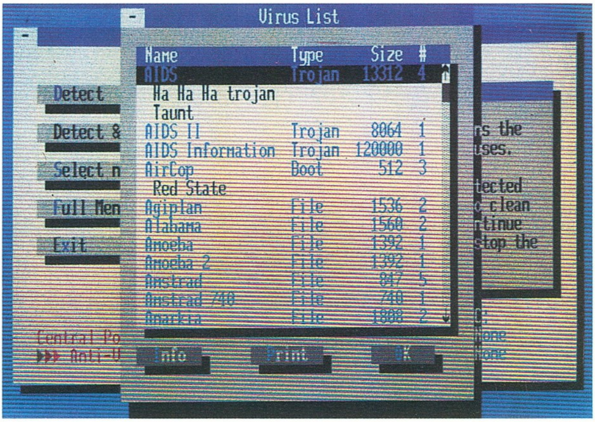
Áttérünk a Central Point Anti-Virus 1.0 ismertetésére. A CPAV már installáláskor is rokonszenves gesztusokkal lopja be magát szívünkbe és gépünkbe. Az Install program előzetes harddiszk- és memóriatesztelést hajt végre, s azonnal kiszűri az esetleges fertőzött állományokat. Készít egy ún. Emergency Disk nevű indítólemezt arra az eshetőségre felkészülve, hogy a bajban jó egy olyan bootlemez, amelyik képes visszaállítani a bootszektort, a partíciós táblát és a CMOS RAM (AT-SETUP) tartalmát, mert az újabb szupervírusok már ide is képesek beköltözni. A BOOTSAFE.EXE program a bootolható meghajtók bootszektorát és partíciós tábláját védi.

Kérhetjük azt is, hogy a CPAV immunizálja, azaz oltsa be a védendő bináris állományainkat az ismert kár- és kórokozók ellen. Két további rezidens program, a VSafe (24K rezidens kód) vagy VWatch (előbbi 8K-s csökkentett méretű változatat) őrködik ébeen rendszerünk háborítatlan működése felett. Minden CPAV program uninstallálható; és jó tudni, hogy nemcsak futtatható programként, hanem a CONFIG.SYS-ben eszközmeghajtóként is betölthető: VSAFE.SYS, VWATCH.SYS.

Szelektív védekezés

Installációkor eldönthetjük, melyek azok a vírusbehatolási módozatok, amelyek ellen fokozottan fel akarjuk magunkat vértezni; esetenként figyelmeztetést kérhetünk:

— HD Low-Level Format: védelem alacsony szintű harddiszkformázás ellen, mely minden adatot menthetetlenül megsemmisít.



— Rezidens módon kilépő, tárban maradó programok betöltése (ez előtti figyelmeztetés).

— Általános írásvédelem (a szuperóvatosaknak).

— Futtatható állományok ellenőrzése (nemcsak betöltés, hanem COPY alatt is).

— Bootszektort támadó vírusok jelenlétének ellenőrzése.

— Harddiszk és floppy bootszektorának és partíciós táblájának (csak HDD-nél) védelme felülírás ellen.

— Figyelmeztetés, ha bárminemű kísérletet észlel, ami egy futtatható állomány kódjának megváltoztatására irányul.

A CPAV a NAV-val ellentétben egy nagy checksum adatbázist tart karban, ezzel komoly helymegtakarítást is elér. Már menteni, azaz backupolni sem engedi a CPAV a fertőzött állományt! A CPAV-nál a futtatható fájlok vizsgálati köre kiterjedtebb, mint a NAV-é: .EXE, .COM, .OVR, OVL, .SYS, .BIN, .APP, .PMG, .PIF, .PRG, .XTP és .CMD.

Nem minden immunis

Az immunizációs eljárás mintegy 1 kilobájttal növeli meg a védendő állomány fizikai méretét, de futtatáskor már természetesen nem jelentkezik ez a memóriatöbblet. A következő állományok nem immunizálhatók:

-- .EXE programok, amelyek overlay vagy debug információkat tartalmaznak kódjuk legvégén.

— Sérült headerrel rendelkező .EXE programok.

— 14 bájtnál kisebb .COM vagy .EXE programok (leszámítva természetesen az .EXE-headert).

— 63 K-nál nagyobb .COM programok.

— Saját önellenőrző kódot tartalmazó programok.

— MS Windows és OS/2 állományok.

(Az immunkód természetesen bármikor levehető a védett állományokról.)

A CPAV kézikönyve végre egy vírusszótárt is magában foglal. A CPAV pillanatnyilag mindösszesen 608 vírust ismer fel név szerint. A Microsoft Windows-ban egy külön erre a célra írt program mutatja meg az éppen rezidensen betöltött CPAV programok státusát. A CPAV Windows-ból is közvetlenül elindítható, látványos gyári ikonok segítségével.

Mindketten a dobogón

Jelen sorok írója nem kíván állást foglalni abban a kérdésben, hogy melyik program jobb a másiknál, helyette egy objektív teszt eredményét adja közre, mely képet adhat a két program víruskeresési sebességéről. A feladat azonos volt mindkét program számára: egy 80 MB-os harddiszk ellenőrzése.

A versenyfutás eredménye a következő lett:

Program	Csak futtatható állományok	Összes állomány
CPAV	45,49 s	2:16,36 perc
NAV 1.5	50,13 s	1:36,73 perc

A döntést a kedves olvasókra és pénztárcájukra bízom. A NAV 15 000, míg a CPAV 16 000 Ft-ért kapható.

Herczeg József

A szerkesztő ajánlata

Munkában megfáradt kollégáknak

A hónap témájához kapcsolódva foglalkozunk a shareware-játékokkal, s mi több, közülük néhányat a ráadásként adott játékmellékletre is rátettünk. Ugye nem haragszanak érte?

A #437 és #438 lemezek számos EGA játékot tartalmaznak, ezek mind önkicsomagoló állományként találhatók a lemezeken.

Tizenkettő két lemezen

Az EGAWHEEL című játék a tv-ből ismert Glücksrad (vagy Wheel of Fortune) számítógépes megfelelője. Közepes angoltudással is kellemes szórakozásban lehet részünk, egyes szólásokhoz azonban elő kell vennünk valamilyen szótárt. Utálatos dolog, hogy ha nem tudjuk kitalálni a feladott szavakat vagy mondásokat, a gép nem mondja meg, mi lett volna a helyes válasz.

Az ALDO játéknál kincset gyűjtögetünk, létrán mászkálunk, adott idő alatt kell a kincseket összegyűjtenünk. Mind EGA-, mind VGA-kártyával játszhatjuk.

A 3DPONG nemcsak EGA-n játszható, hanem CGA-val is, vezérlése történhet billentyűzetről, joystickkel vagy egérrel. 9 sebességi szinten, 4-féle nehézségi fokkal játszhatjuk. Mint nevéből is kitűnik, 3 dimenzióban pattan vissza a labda a falakról, az a cél, hogy minél többször visszaüssük. Nem könnyű a feladat.

A BACKGAM EGA- és VGA- kártyákkal működik, vezérléséhez Microsoft egér szükséges. Három nehézségi szint közül választhatunk, a harmadikat azonban csak akkor érdemes kipróbálnunk, ha a másodikon már folyamatosan nyertünk.

A QUINTA-t is játszhatjuk mind EGA-, mind VGA-módban. A játék célja az, hogy öt lapkát egymás mellé helyezzünk (vízszintesen, függőlegesen vagy ferdén). Két játékos is játszhatja vagy egy játékos a géppel, vagy a gép önmagával. Egér itt is feltétlenül szükséges.

A TRISK tulajdonképpen az ismert Rubik-gyűrűk forgatásának számítógépes változata. EGA és 124 K memória szükséges. Az 1, 2, 3 billentyűkkel az óramutató járásának irányába forgatunk, míg a 4, 5, 6-tal ezzel ellentétesen. Mivel a három gyűrű egymásba kapcsolódik, a forgatások nemcsak egyet-egyet érintenek, ez természetesen nehezíti a megoldást. Nagyon kedves ötlet volt a szerzőktől az, hogy számos más játékkal ellentétben itt bármikor problémamentesen kiléphetünk egy Q(uit) begépelésével, ha úgy látjuk, hogy már végképp reménytelen a helyzetünk.

Az EGATETR egy nehezített kétdimenziós TETRIS-változat. 256-féle játékvariánsból választhatunk. EGA-kártya és 256 K memória szükséges. A játék során a J, K, L billentyűk és a szóköz leütésével irányíthatjuk az egyes alakzatokat.

Az ARKEGA az ismert falbontó játék EGA-változata. Szerzője elégedetlen volt az Arknoid PC-változatával, ezért elsősorban önmagának készítette el ezt a játékot. Egérrel és billentyűzetről is vezérelhető. A háttér színe fokozatonként más és más, gyönyörűek a minták. Nagyon gyors, nehéz játszani, ha elfelejtjük kikapcsolni a Turbo módot!! A leírás értelmes, tömör és igaz.

Az EGASOLIT olyan pasziánsz, amelynél alapértelmezésben minden harmadik lapot nézhetünk meg, de ha ez nem tetszik, módosíthatjuk a szabályt: egyesével minden kártyát megnézhetünk, akár végtelenítve a pakli fordítását, akár pedig csak egyetlen körfordulattal. Alapértelmezést választva akárhányszor forgathatjuk a paklit. A játék vezérlése egérrel történik.

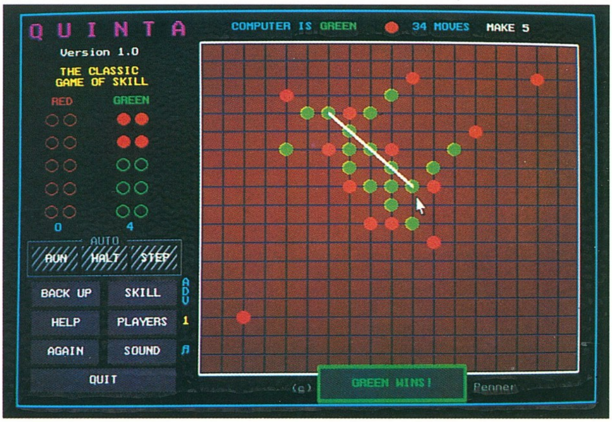
A FANS tulajdonképpen egy TETRIS-szerű ötleten alapszik, megspékelték némi lövöldözéssel, jutalmunk aprópénzeső lehet.

A KLONDIKE kártyajáték, mind EGA-, mind VGA-kártyával működik, Microsoft egérrel, illetve billentyűzetről vezérelhető. Szerencsés, hogy a pasziánsz szabályai módosíthatók. Ez azért is fontos, mert a szerzők bevallása szerint is csak minden harmincadik játékban nyerhetnénk, ez pedig bárkinek elvenné a kedvét a játéktól.

Az OTHELLO amőbaszerű játék, amelyet ketten is játszhatnak, illetve a gép lehet az ellenfelünk. Elég nehéz a géppel szemben nyernünk.

Fránya pecabotok

A #158-as lemezen található pecázóbajnokság viszont kifogott rajtunk. Sok-



sok türelemre lenne szükség ahhoz, hogy az összes szabályt betartsuk, és még halat is fogjunk. Csakis úgy érdemes leülni a játék elé, ha hosszabb időt szánunk a szabályok tanulmányozására. Máskülönben folyton büntetéseket szabnak ki ránk olyan vétségekért például, hogy csak úgy mentőmellény nélkül merészkedtünk a nyílt vízre. Az alapos angoltudás sem elegendő, legalább egyszer a nagyszótárat is át kell rágnunk, máskülönben csak találomra választhatunk a millióféle horgászbot és egyéb kellék közül. Mivel a játék grafikája sem valami jól sikerült, be kell vallanunk, hogy néhány órás hiábavaló próbálkozás után feladtuk. Számunkra már soha nem derül ki, hogy vannak-e szebb képernyők, és ki mekkora halat tud kifogni.

A szerkesztő kedvencei

Nagyon szubjektív vélemény ugyan, de a #467-es lemez az egyik kedvencem. Bárki látta, valahogy jobb kedvre derült tőle, ez manapság nem kis eredmény. A játékot 4—8 éveseknek ajánlják, de zárójelbe teszik a 80-at is felső korhatárként. Így igaz. Mindenki lelkesedett a jutalmul kapott nevetős kismackókért, a kétéves kislánytól gimnazista nagyfiúkig. Az utolsó logikai feladatnál pedig (itt egy teherautóra kell felrakni megadott sorrendben alakzatokat) az összes gyerek hamarabb rájött a megoldás menetére, mint a számítástechnikában megfáradt szülők. Biztos, hogy nemcsak a szórakoztatás volt a programkészítők fő célja, de tényleg szemgyönyörködtetőre sikerült a grafika és az egész játék, zenéje is, bűbájos. Taníthatjuk vele tízig számolni a gyerekeket, a logikai feladatokat is biztos szívesen oldják meg. A betűknél azonban sajnos nehéz lenne a kicsiknek megmagyarázni, miért kell a G (Girl) betűt a kislányfigurához rendelnünk. Az angol nyelvvel ismerkedő gyerekek valószínűleg örülnek, ha néhány olyan szóra bukkannak, amit már ők is ismernek.

A #344-es lemezről indul a szerelmes DOS, egy nagyon kedves rendszer. Érdemes azonban előzőleg elolvasnunk a leírást, máskülönben esetleg csak Resettel tudunk kilépni a magunkra szabadított szerelmes DOS-ból. Ha kollégáink elég fejlett humorérzékkel rendelkeznek, és idegrendszerük is viszonylag jó állapotban van, betehetjük a LoveDOS-indítást egyik este az AUTOEXEC-be. Reggel pedig vagy megvernek, vagy igen kellemesen indul mindannyiunk munkanapja.

Verebély Pálné

Solarsoft játékok sikerlistája

(A Floppyland ez évi eladásai alapján)

#466 Skyglobe Star Gaser
#356 VGA Games #2
#463 Games for MS Windows
#437 (és #438) EGA Games #1 és #2
#350 VGA Games #1
#233 Greatest Arcade
#255 Crossword Creator
#440 Chess Games
#267 Arcade Games
#179 Dracula in London
#258 Mahjongg
#351 Captain Comics
#439 Bridge, Canasta
#232 ASCII Arcade
#349 Springer
#189 Lotto Master

SolarSoft sikerlista

(Az 1991. júliusi és augusztusi eladások alapján)

	No.	Programnév	Db	Programleírás
1.	510	ARJ 2.10	1	A PKZIP/LHA-nál is hatékonyabb sláger adattömörítő!
2.	466	SKYGLOBE STAR GAZER	1	Színes, mozgó csillagtérkép amatőr csillagászoknak
3.	475	NEWSPACE	1	A harddiszk kapacitását megkétszerezi!
4.	421	PKZ110 & ZIPDMP & SHEZ	1	A „sűrítés" magasiskolája és Norton Commandere
5.	319	SCAN80 & OTHERS	1	McAfee-féle vírusmegelőző, -detektor és -ölő
6.	432	LZEXE & LIST 7.5e	1	Gyors EXE kompresszor, Vernon Buerg LIST PLUS-a
7.	468	SUPER ASSEMBLER ED.	1	Turbo Assemblerre kihegyezett programeditor
8.	509	ZIPVIEW & LHA 2.10	1	Az LHARC továbbfejlesztése és egy NC-kiegészítés!
9.	474	JORJ POP-UP DICT.	2	58.000 szavas tárrezidens angol értelmező szótár
10.	383	4DOS V3.02	1	COMMAND.COM-pótló DOS-héj: 50 új parancs
11.	423	QFont 1.15b	1	Szoftfonteditor magyar Venturához!!!
12.	096	AS-EASY-AS 4.00p	1	Lotus-kompatibilis táblázatkezelő, egyszerűbb
13.	508	PROCUBE LITE 1.0	1	A QUBECALC 3D táblázatkezelő továbbfejlesztése
14.	470	MULTI-EDIT 5.0	1	A világon legjobbnak tartott programozói editor
15.	435	OPTIKS & ICONVERT	1	PCX, PIC, GIF, TIF, GEM, MAC... grafikus konverterek
16.	304	TURBO TECHNO JOCKS	2	Szuper Turbo Pascal-unitok forrásaikkal!
17.	M020	PC-JOG	2HD	Hatályos jogszabályok visszakereső rendszere
18.	442	WINDOW PRO 1.51	1	No.1 ablaktechnika C nyelven (small/medium/large LIB)
19.	484	SR-INFO	2	dBase-kompatibilis fejlesztőrendszer!
20.	427	ANADISK, CON-FMT	1	Lemez-, fájl- és FAT-editor, rezidens formattáló!
21.	507	PC-BROWSE & NG.	1	Szenzációs hipertext rendszer és egy NG decompiler
22.	441	DATABASE IN C	1	Adatbázis-kezelés C-ben, forráskóddal (Btree, dBASE)
23.	505	GALAXY LITE 1.6	1	A nagy sikerű GALAXY editor felújítása
24.	329	PC-MAGAZINE BENCH.	1	Az USA -szaklap hardvertesztjei szervizeseknek (v5.0)
25.	422	FONTEDIT 5.7	1	HP-lézernyomtatókhoz szoftfontok
26.	070	BLACK MAGIC	3	Grafikus módú hipertext, önálló futtatómodullal
27.	446	C-MIX #4	1	Online változtatható méretű RAMDISK EMS/XMS-ben
28.	384	HEXCALIBUR	1	RAM-editor blokkműveletekkel (insert/delete/move)
29.	504	PC-TYPE + 2.0	3	Szuper szövegszerkesztő program, grafikát is kezel!
30.	506	PROLITE 1.01	1	A PC-WRITE szövegszerkesztő egyszerűsített változata

Multi-Edit 5.0

Szinte túl jó!

Az Alaplap ez évi júliusi számában mutattuk be az ASMED nevű programeditort, amely leginkább assembler nyelvű programok fejlesztésénél bizonyul hasznos segítségnek. A Solarsoft programkönyvtárban #470-es sorszámmal megtalálhatjuk a Multi-Edit program shareware-változatát, amely — sok más egyéb mellett — a magas szintű programnyelveken történő programfejlesztést támogatja. (A program kereskedelmi változata is beszerezhető Magyarországon.)

Miért is vettük elő a Multi-Edit 5.0 programot tesztelésre? Azért, mert a programban nemcsak kiváló ASCII szövegszerkesztőt, hanem teljes programfejlesztő környezetet találtunk, amellyel programjaink mellett a leveleinket is megírhatjuk. Azért, mert a nagyszerű program kereskedelmi verziójának megvásárlása előtt a némi korlátozásokkal működő shareware-változattal csaknem teljes egészében kipróbálhatjuk, mit is fogunk kapni a pénzünkért.

Rövid tanulás és gyakorlás után — amihez bőségesen elegendő a beépített helyzetérzékeny help rendszer — bárki megtalálhatja a Multi-Edit programban azt a társat, amelyet már hosszú ideje keresett.

Nézzük részletesebben a program szolgáltatásait, tulajdonságait! A Multi-Edit parancsait 3 módon adhatjuk ki, a funkció- és gyorsbillentyűk segítségével, a redőnymenüből kiválasztva, valamint egérrel.

Eldönthetjük, hogy az egyre inkább elterjedő SAA-szabvány szerinti vagy inkább a Multi-Edit saját, a szerzők által egyszerűbbnek tartott kezelői felületét szeretnénk használni. Hiányosságnak tartom viszont, hogy a felső sor redőnymenüjének pontjai nem érhetők el az Alt és a kiemelt kezdőbetű együttes leütésével, a programban szerkesztés közben az F2 vagy az Esc billentyű hatására lesz aktív a főmenü.

Dicséretes módon jól kezelhető a program help rendszere. Betöltés után azonnal szembetűnik az alsó sorban a tíz funkcióbillentyű jelentésének rövid magyarázata. A Shift, a Ctrl vagy az Alt billentyűk lenyomására azonnal előtűnnek a módosított jelentések magyarázatai. Az F1 billentyű hatására a program megjeleníti a segítőrendszer tartalomjegyzékét, ahol kiválasztható a keresett téma. A teljes help rendszer fastruktúrája bejárható a szintek közötti lépegetéssel — pillanatnyi tartózkodási helyünket is mindig látjuk —, azonban a hipertext rendszerekhez hasonlóan a program minden témához felkínálja a kapcsolódó témákhoz történő átlépés lehetőségét.

A programhoz több segítő állományt is mellékeltek, ezek bármikor betölthetők, tanulmányozhatók. A help képernyőn megjelölhetők területek, amelyek néhány billentyűleütéssel átmásolhatók az éppen szerkesztett szövegbe. A felhasználó bővítheti az asszociatív help rendszert, valamint megszerkesztheti saját menüjét is a gyakran használt parancskombinációkkal és állománynevekkel.

A Multi-Edit egérrel is használható. A Windows operációs rendszer használata során megszokott módon nagyíthatók, mozgathatók az ablakok, illetve navigálhatjuk az aktuális pozíciót az állományon belül. Meglepően barátságosan működik a program, szöveg beírásakor más szerkesztőprogramok használata során sokszor az első mozdulatom az egér félretolása volt, hogy a képernyő közepén állva ne zavarjon az oldal áttekintésében. A Multi-Edit egere az első billentyű leütését követően — pozícióját megőrizve — eltűnik a képernyőről, az egér elmozdítását követően válik ismét láthatóvá. A program ablakrendszere világos, egyszerűen kezelhető. Egyidejűleg 100 ablak lehet megnyitva, ezek nagysága egyszerűen beállítható. Minden ablak egy billentyű leütésére teljes képernyőnyire nagyítható, majd visszaállítható az eredeti méret. Az ablakok ikonná alakíthatók, ekkor mindössze a szerkesztett állomány nevét látjuk a képernyőn. Bár az ablakok keretei, a státussor és a főmenü sora egyaránt eltávolíthatók a képernyőről, gondolom, nem én vagyok az egyetlen felhasználó, aki örökösen információhiányban szenved, ezért ezek mindegyikét a képernyőn tartom. Ezek száma egy ablak esetén 5, minden újabb vízszintes ablak megnyitásakor eggyel növekszik. A hagyományos 25 soros képernyőn hamar úgy találjuk, hogy több a háttérinformáció, mint a szerkesztett szöveg. Ezért ha tehetjük (EGA vagy VGA videorendszer és megfelelő láthatóság esetén) érdemes átkapcsolni 43/50 soros üzemmódra, és máris kitágul előttünk a világ. Az egyes ablakokban különböző állományok vagy egy állomány különböző részei szerkeszthetők. A szerkeszthető állományok legnagyobb mérete 32 Mbájt vagy 2 milliárd sor lehet, ennél nagyobb szöveges állományokkal viszonylag ritkán találkozunk.

A <File> menüben található a <DOS Directory shell> parancs, ami önálló programként is megállná a helyét. Segítségével tetszőleges állományokat másolhatunk, törölhetünk, átnevezhetünk, nyomtathatunk, sorba rendezhetünk. Több ablak is megnyitható egyidejűleg, ezekben az állományok listáját vagy a grafikus könyvtárfát tekinthetjük meg. A programból ideiglenesen ki is léphetünk az operációs rendszerbe, ekkor a Multi-Edit mindössze 2 kbájtot foglal le az operatív memóriából.

A szinte korlátlan lépésszámú (be kell állítani, alapértelmezés szerint 100 lépésig terjed az emlékezete!) Undo funkció mellett az automatikus állománykiírási funkció is „életmentő" lehet néhány esetben. Ez utóbbi használatakor meg kell szabni annak az időnek a hosszát, ami alatt ha nem történt billentyűleütés, a program automatikusan kimenti az állományt.

A Multi-Edit fontosabb adatai

— A szerkeszthető állományok legnagyobb mérete 32 Mbájt vagy 2 milliárd sor lehet, 2048 oszlop kezelhető
— Egyidejűleg 100 állomány szerkeszthető
— Helyzetérzékeny, bárhonnan meghívható help rendszer
— A Windows használata során megszokott egérkezelés
— 65 535 lépésig működő Undo és Redo (az Undo Undója) funkció
— Kereséskor reguláris kifejezések használhatók
— Egyidejű keresés több állományban
— Bármikor előhívható ASCII-tábla és kalkulátor
— Programnyelvekhez illeszkedő automatikus indentálás
— A fordítóprogram elindítható a Multi-Editből való kilépés nélkül
— Automatikus fordításhiba-pozicionálás
— EMS-használat, a nagyméretű állományok is kezelhetők lemezműveletek nélkül
— Nagyméretű programok fordításakor mindössze 2 kbájt erejéig marad a memóriában a Multi-Edit 5.0
— Opcionális kommunikációs, helyesírás-ellenőrző, makrófejlesztő modulok
— A felhasználó megszerkesztheti saját menüjét a gyakran használt parancskombinációkkal, nevekkel
— Felhasználó által bővíthető asszociatív help rendszer
— Teljes DOS-keretrendszer a programon belül

Olykor hasznos lehet a vonalzó használata, a <Ruler> paranccsal elővéve tetszőlegesen mozgatható a képernyőn, mérhetünk vele, majd ha már nincs rá szükség, eltüntethető.

A Multi-Edit kiváló makrókészítési és -futtatási lehetőségekkel rendelkezik. Tetszőleges billentyűsorozat hozzárendelhető egy billentyűhöz, ezekből jó néhány darabot készíthetünk.

A blokkműveletekkel mindent elvégezhetünk, amire szükség lehet. Kijelölhetünk sorokból, oszlopokból, valamint karakterfolyamból álló blokkokat. A blokk törölhető, másolható, mozgatható, akár különböző ablakok között is. A kijelölt blokk egy átmeneti tárolóba (scrap, clipboard) helyezhető, ahonnan később bármikor elővehető. Nagyon hasznos az átmeneti tár tartalmához történő hozzáadás lehetősége.

Keresés és csere során reguláris kifejezések is használhatók, amelyek sok esetben megkönnyítik feladatunkat, azonban elengedhetetlen, hogy az opció kikapcsolható legyen. Nos, a Multi-Edittel ezt is megtehetjük. Az <Undo> parancs természetesen az egy lépésben kicserélt nagyszámú karaktersorozat visszaállítását is elvégzi, méghozzá egy műveletnek tekintve. Egyidejűleg több ablakban és állományban is kereshetünk. A szerkesztett állomány kiterjesztésétől függő módon végzi az automatikus indentálást a Multi-Edit. Minden ismert programnyelvhez (a shareware verzióban csak Pascal és a C nyelvhez) elkészítették az úgynevezett Template leírást, ami alapján az ismert kulcsszavak kifejtése is automatikus. Például .PAS kiterjesztésű állomány szerkesztésekor a <p> és az <Alt-F9> billentyűk leütésére a képernyőn a következőket látjuk:

```
procedure ;
  begin
  end;
```

A kurzor természetesen a pontosvessző előtt áll, hogy az eljárás nevét beírhassuk. Minden programnyelvhez definiálható a fordítóprogram, és annak parancssora, amivel az éppen fejlesztés alatt álló programunkat az editorból való kilépés nélkül lefordíthatjuk. Szükség esetén a Multi-Edit program kilapozza önmagát EMS memóriába vagy mágneslemezre, hogy a nagy memóriaigényű fordítóprogramok is elindíthatók legyenek. Az esetleges fordítási hibák sorára automatikusan rááll a program, jelezve a fordítási hibaüzenetet.

A Multi-Edit program nemcsak programeditorként, hanem dokumentumszerkesztőként is használható. Ily minőségében persze nem hasonlítható a Microsoft Wordhöz vagy a WordPerfecthez, mindazonáltal a fontos funkciók mindegyikét megtalálhatjuk a Multi-Editben is. Beállíthatók a karaktertípusok, fejlécet, lábrészt szerkeszthetünk, automatikusan készíthetünk tartalomjegyzéket, az oldalak mérete, a tabulátorpozíciók is szabadon állíthatók. Sajnos a nyomtatóvezérlés csak a kereskedelmi verzióban működik.

A Multi-Edit program bármikor előhívható teljes ASCII táblával, valamint több számrendszerben működő kalkulátorral könnyíti munkánkat. Ez utóbbiból a számítás eredménye azonnal bemásolható a szerkesztett állományba.

A vonalrajzoló paranccsal egyszerűen szerkeszthetünk szimpla és dupla vonalas táblázatokat, egyszerűbb vonalas ábrákat.

Az opciók és paraméterek a programon belül állíthatók be, és a kereskedelmi verzióban el is menthetők. A shareware változat talán legnagyobb korlátozása az, hogy a program pillanatnyi állapota nem tárolható el.

A Multi-Edit kereskedelmi verzióját még hatékonyabbá teszi a magas szintű makronyelv, amellyel tetszőleges új funkciót definiálhatunk. Beépítették a makródebuggert is, amellyel saját makróink esetleges hibás működését nyomon követhetjük.

A program szokatlan, de néha igen hasznos tulajdonsága a megváltoztatott szövegsorok megjelölése. Egy szövegállomány betöltése után a sorok fehér színűek. Ha bármelyik sort megváltoztatjuk, vagy új sort írunk be, azokat dupla intenzitással látjuk, így igazán könnyedén nyomon követhető, hogy mit is változtattunk a szövegen.

A beolvasott állományok a véletlen felülírástól megvédhetők a „read-only" attribútum beállításával, ekkor nyugodtan böngészhetünk az állományokban, garantáltan nem tehetünk kárt bennük.

A kereskedelmi verzióban kommunikációs modult, terminálemulációt, valamint angol nyelvű helyesírás-ellenőrző modult is találhatunk, mellyel külföldi levelezéseinket ellenőrizhetjük.

A program együttműködik a DeskView, a Windows 3.x rendszerekkel, valamint OS/2 alatt is használható. A CWI-kód szerinti ékezetes magyar karakterek gond nélkül alkalmazhatók a szerkesztett szövegben, jelen írás is a Multi-Edit programmal készült.

Lóth Tamás

Mi az újdonság az ARJ 2.20 verziójában?

SOK!

Az ARJ az 1991-es év sikerprogramja. A SolarSoft shareware lemezei között májusi keltezésű 2.10-es verzió hosszú ideje vezet az eladási listán. A szerző, Robert Jung talált azonban néhány javítanivalót, így az új, 2.20-as változatban vélhetően már alig néhány hiba maradt. Mi mindenesetre közreadjuk a javítások és bővítések (szerintünk) teljes listáját.

— A /ji<indexfile> kapcsoló (= indexfájlt készít) használatakor az esetleg már meglévő indexfájl nem vész el, hanem az újabb adatokat a meglevő fájl végére illeszti a program.

— A w (= Where_is_?) parancs második userpromptját megváltoztatták. Eddig azt kérdezte, mutassa-e a megtalált string környezetét, most pedig azt kérdezi, hány sort (0—24) mutasson be belőle.

— Az UNARJ220.EXE immár levált a törzsállományról, külön állományba került.

— Gyorsult az SFX modul. Most gyorsabb, mint az ARJ x.

— Kibontáskor az e és x parancsok használata esetén (= EXtraxt) csak egyszer írja ki a kibontott fájlok nevét, ha nem adunk új nevet a kibontott állományoknak.

— Frissítéskor és aktualizáláskor (f = Freshen és u = Update parancsok) csak akkor készít új ARJ fájlt, ha van mit bepakolni.

— Az o parancs (= Order) mellett megáll a program, ha többrészes ARJ fájl kerül elé.

— A p parancs (= Print) mellett a /jp (= Pause) kapcsoló a DOS MORE szűrőparancsához hasonlóan megállítja a kiírást, ha egy-egy képernyő betelik.

— Az ARJ_SW használatakor a változóhoz most már nemcsak közvetlenül a kapcsolókat tartalmazó stringet rendelhetjük, hanem egy konfigurációs fájlt is. Ha a változóhoz rendelt szöveg / vagy — karakterrel kezdődik, akkor az ARJ a megadott kapcsolókkal az összes parancsot kiegészíti alapértelmezésként. Ha azonban nem a fenti bevezető karakterekkel kezdődik az ARJ_SW-hez rendelt string, akkor azt a program egy konfigurációs fájl neveként értelmezi. Az így megadott konfigurációs fájlra a következők érvényesek:

- Közönséges ASCII textfájl.
- Tetszőleges számú parancs kapcsolóinak alapértelmezését módosíthatjuk a konfigurációs fájl segítségével.
- Ahány parancsot módosítunk, annyi sor kell.
- Az egyes sorok formátuma: a sor legelejére a parancsot kell írni, a kapcsolókat pedig utána:
<parancs> /<kapcsoló> [... /<kapcsoló>]
vagy
<parancs> -<kapcsoló> [... /<kapcsoló>]
- A konfigurációs fájlban csak egyféle bevezető karaktert használhatunk egyszerre, vagy a / vagy a — jelet.
- Ne legyenek benne megjegyzések és üres sorok.

— Az ARJ 2.20 verziótól a program elfogadja, ha az ARJ_SW-ben és a parancssorban nem azonos kapcsoló bevezető karaktereket használunk. A 2.10 verzióban ilyenkor hibaüzenettel leállt a program.

— A /$ kapcsoló most már nem ír ki figyelmeztetést, és az ERRORLEVEL-t sem állítja át, ha nincs elmentendő lemeznév.

— Az /a és az /a1 kapcsolókkal most már helyesen kezeli le a rejtett könyvtárakat.

— A 2.20 verziótól a /d kapcsoló (= Delete) a j (= Join) parancs mellett is alkalmazható.

— Az /m3 és az /m4 kapcsolókkal az ARJ 2.20 gyorsabban és valamivel tömörebben dolgozik, mint a 2.10 verziónál. Az /m4 kapcsoló most majdnem kétszer olyan gyors tömörítést eredményez, mint az /m1 kapcsolóval, ami az alapértelmezés.

— A /vv kapcsoló alkalmazásakor a többrészes archív utolsó részének elengedésekor hangjelzést ad (sípol).

— Kommentezni az ARJ fájlt most már a bepakolással, módosítással egy időben is lehet, nemcsak a c (= Comment) paranccsal. A /z kapcsoló a 2.20 verzióban már használható az a (= Add), u (= Update), m (= Move), f (= Freshen), j (= Join), r (= Remove path), d (= Delete), n (= reName), o (= Order) parancsokkal. Csak az e, x, l, v, t, p és s parancsokra nincs hatása.

— A /z mellett már a parancssorból is adhatunk kommentfájlt. Ilyenkor nem kell az ARJ-nél már megszokott ! listafájljelet a fájlnév elé írnunk.

— A /jl<listafile_név> kapcsolóval játszva készíttethetünk az ARJ-vel listafájlokat. Ilyenkor csak a fájlnevek kerülnek ki a <listafile> soraiba, soronként egy-egy.

— Egy új kapcsoló a /jw<file_név>. Ez lehetővé teszi, hogy egyes fájlokat ne az archívban viselt névvel bontsunk ki.

— A /t1 kapcsoló a leírás szerint most már korrektül viselkedik akkor is, ha nem ASCII textfájl kerül elé.

— Az ARJ-SFX modulhoz egy belső azonosító stringet adtak hozzá: "aRJsfx". Ez az első kb. 100 bájton belül van.

— A /va kapcsoló alkalmazásakor a szabad lemezterület ellenőrzését is elvégzi a program.

— Megnövelték a kibontás sebességét.

— A /jx kapcsolóval megadhatjuk, hogy kibontáskor a kimeneti fájl mely pozíciójában kezdjen el írni.

— A /jy kapcsolóval elérhetjük, hogy a (Yes/No) válaszoknál elég egy betűt lenyomnunk, és a billentyűpuffert is kiüríti a válaszkérés előtt a program.

— Külön számolja a fájlokat és a kommenteket.

— Az a /n paranccsal a 2.10 verzióban még előforduló hibát kijavították.

— Az "ARJ o "/ARJSORT alkalmazásnál is kijavítottak egy hibát.

Nagy Gábor

A legelterjedtebb számítógépekre

Anyanyelvi programnyelv kezdőknek

A számítástechnika alapnyelve az angol. Hány milliós bevételkiesést jelenthet ez a számítástechnikai iparnak, a forgalmazóknak, és mi lenne a megoldás? A PC Turbo Klub szeptember végi összejövetelének előadója érdekes példával szolgált.

— Láttam én már számítógépet, hasznos is lehetne a munkámhoz. De miért venném meg, ha nem értem, mit ír ki a képernyőre — hangzik angolul nem beszélő, de racionális, a saját szakterületükben otthonos vállalkozók szájából. Azokéból, akik valóban ezért nem fordulnak segítségért a számítógéphez.

— Nem tudom, mit akar a tanár úr, még magyarul sem mindig értem, mit magyaráz, pláne „külföldiül"! — vetődik fel a diákban, akit a számítástechnika még nem igézett meg.

Az angol nyelvterületen élők, ha meglátják egy program képernyőjét, vagy a programlista szövegét, az abban szereplő szavak jelentését azonnal megértik. Ezt az előnyt miért csak az angol anyanyelvűek élvezzék, és más anyanyelvűek nem? Aki meg akar tanulni a számítógépek nyelvén, az tanuljon meg angolul, hangzik a sokszor hallott felelet. Kötelező megtanulni! Megint valami, ami kötelező. Legyen választható, legyen előbb megszokható a gondolkodásmód. Utána a nyelvezet lefordítása igényként is felmerül, ha más országbeliek programjait megérteni, használni akarja az újdonsült felhasználó.

Főleg, ha ebben a program támogatja. Gombnyomásra lefordítja anyanyelvéről a szabványosra, angolra. Gyerekek és felnőttek oktatására egyaránt kínálkozó lehetőség.

Ha egy forgalmazótól az adott kínálati feltételek között nem vásárolnak több számítástechnikai terméket, akkor változtat. Valami újszerűt kell bevetni. Szélesebb kínálatot kell tárni a vásárló elé, amiből szívesebben választ. Ez jelentheti például a különféle felhasználói programok anyanyelvűvé tételét. Itthon is megvalósult már, hogy a külföldi nagyvállalat termékeit forgalmazó cégek betársultak a fejlesztői munkába, az adott szoftver honosítását valósítják meg. A német nyelvterületen általános követelmény a honosított szoftverkínálat. A nagy német felvevőpiac biztosítja az ilyen vállalkozások sikerét, hiszen biztos a nagy darabszám, és biztos a siker. Ez így van akkor is, ha viszonylag szűk területet sikerül lefedni. A magyar nyelvterület kicsinysége szigorúbb feltételeket szab.

Döntő szempont, hogy nagy darabszámú gépen, széles körben igényeljék a felhasználók a program szolgáltatásait. Mi lenne, ha egyszerűen minden gépen futna a program? Egyetlen kifejlesztett program! A megoldást a különféle szimulációk és emulációk adják.

Egy slágergép szimulációi

Egy bizonyos gépet, annak működését egy hozzá alapvetően hasonló, például azonos mikroprocesszorú vagy egy nagyobb számítógép tudja helyettesíteni. A szoftver vagy hardver, vagy a kettő kombinálásával megvalósított szimulációk, emulációk céljai eltérőek lehetnek. Az azonos kategóriájú, hasonló gépeknél a szoftverek bővítése lehet a cél a másik gép programjaival. A fejlettebb gépeknél — mintegy keresztfejlesztői környezetként — a fejlesztési lehetőségek bővítése indokolja a szimulációt.

A szimulációkhoz kialakulhatnak népszerű rendszerek. Hogy mi határozhatja meg egy gép slágerré válását, azt pontosan nem lehet megmondani. Talán, hogy hamarabb került a piacra, és hatékonyra, barátságosra sikerült termék, és az előzők miatt kiemelkedően jó a szoftverellátottsága. A mi esetünkben a slágergép, amelynek szimulációját gyakorlatilag az összes Magyarországon elterjedt számítógépen megvalósították, a jó öreg Sinclair ZX Spectrum. A kompatibilitást kisebb-nagyobb mértékben biztosító számítógépek az azonos Z80 processzorú Enterprise és a Videoton Home Computer. A HT 3080 C és a Primo felépítése szintén lehetővé tette a különböző módon megvalósított részleges szimulációt. Az emuláció a periferiális, I/O különbségeket hivatott eltüntetni — mint a billentyűzet figyelése, portok átcímzése, a képernyő utánzása. Hardverúton kiegészítő áramkörök, szoftverúton az alapprogram megfelelő részeinek átírása nyújt kielégítő megoldást.

A más processzorú gépeken a gépek teljesítőképességét maximálisan kihasználó szimulációkat igyekeztek létrehozni. A C64-re elég korán, még 1985-ben elkészült szimulátor a ZX BASIC-jét valósította meg, a lehetőségekhez mérten maximális helyet hagyva a programoknak. A ZX ROM programját a 6510-es mikroprocesszor utasításaira fordították át, ráadásul az Interface 1-gyel bővített ZX funkcióiból is beépítettek néhányat. A Commodore floppyját a ZX microdrive-kezelő utasításaival lehet elérni. Értelemszerűen a Z80-as gépi kódú program végrehajtására való USR utasításnál hibaüzenettel leáll az interpreter.

Az IBM PC/AT-kompatibilis gépekre régóta vannak crossassemblerek a Z80 felé (X80), léteznek Z80 emulátorok, sőt CP/M-emulátor is (Z80MU). Nem volt meglepő tehát, inkább váratlan a Prágai Műszaki Egyetem Villamosmérnöki Karát származási helyként feltüntető ZX Spectrumszimulátor megjelenése a nemzetközi BBS-eken. A program egy Z80 debuggert magában foglaló, gépi kódot is futtatni képes, a programtárolást „kazettának látott" fájlokon megoldó szimulátor.

A szimuláció foka a nehézségek, a megoldhatatlan problémák, a kompromisszumok, a kitűzött cél, az ár/teljesítmény arányaival mérhető. A szoftverszimuláció gyors, olcsó, kiegészítő hardver nélküli megoldás. A hardverszimuláció jobb eredményt adhat, de a gyártás megszervezése, az alkatrészek, a fröccsöntő szerszám mind drágítják. Konkrétan a ZX-szimulációkban a képernyő megvalósítása tűnt a legnehezebb feladatnak. Ha megelégszünk a fekete-fehér üzemmóddal és a ZX standard

Basicjével, *az összes felsorolt gépen azonos futási eredményt kapunk!*

Szerencsére a mi igényeinket gyakorlatilag teljesen kielégítik — a kompromisszumokon belül is — a létrejött szimulációk. Így akár a legtávolabbi iskolában is elérhető legalább egy gép.

Az anyanyelvi Basic

Az angol nyelvterületen élő érti a programok listáit. De már a magyar anyanyelvű, angolul (még) nem beszélő is. Lengyel, szlovák, cseh, román, orosz, ukrán, bolgár, német, olasz, és bármelyik nyelvre átírható az anyanyelvi Basic kulcsszótáblázata.

A program a Sinclair ZX Spectrum 16k/48k bekapcsolás utáni Basicjének minden kiírását, a hibaüzeneteket is magyarul adja. Akár a ZX-be, akár bármelyik ZX szoftverszimulátorba vagy hardveremulátorba töltődik be, ugyanazon kulcsszavakat írja ki. Az anyanyelvi üzemmódból angol üzemmódba, a C64 kivételével, egyetlen utasítássor begépelésével juthatunk. Ez visszafelé is érvényes.

Az anyanyelvi Basic — gyerekek és az idegenkedő felnőttek számára — talán az első lépcsőfok lehet a számítógépek felé. Talán ennél nem is több.

A program néhány éve, eredetileg csak ZX Spectrumra készült. Elindítása után a gép az eredetivel azonosan működik. Eltérés, hogy a NEW parancs nem törli a teljes Basic-területet, csak az illetékes rendszerváltozókat állítja át, továbbá a szerkesztő üzemmódban másféle hangot ad a billentyűnyomásra, mint input üzemmódban.

A program a ROM program alapján készült. Az eredeti főprogramot a RAM-ban újraírva a többi szubrutint másképp vagy más bemenőadattal hívjuk meg. A módosítások biztosítják az anyanyelvi működést. Ugyanis a program — mielőtt a hibaüzenetet vagy az anyanyelvi utasítást kiírná — a karaktergenerátor kezdőcímét átírja a mi karakterkészletünkre, majd visszaállítja az addig használatos karaktereket. Akár cirill, akár ékezetes, akár más írásjelekre.

A szükséges változtatások a hibaüzenetek és a kulcsszavak kiírásához kellenek. Az átírt NEW rutin nem adja vissza a vezérlést az eredeti ROM programnak, mert az alaphelyzetbe hozná a gépet. Ismét be kellene tölteni a programot. Emiatt a futtató rutinok jelentős részét is újra kellett írni, és az utasítások címtáblázatát a RAM-ba másolni, ahol a NEW utasítás a módosított címét kapja meg. A kulcsszavak átírásához elegendőnek tűnt a kiíratások rutincímét átírni a RAM-ban, de így csak az első CLS-ig működne, mert az visszaírja. Tehát a CLS rutin vezérlését is át kellett programozni, és az összes olyan rutint, amely a CLS-t, vagy ennek egy részét hívja. Az EDITOR rutin egy részét is ezért kellett átformálni. Ebbe került be a gombnyomást visszajelző hang helyére egy érdekesebb hangeffektus. A program nem ír be a ROM területére, a szabványos be- és kilépési címeket használja. Így az ezekre kényes, Z80 alapú vagy emulálóprogramokon ZX-szimulátorokon fut.

A program a C64-en a ZX-től eltérő módon installálódik. A szimulációs program RAM-ban fut. Így a kulcsszavak átírhatók. A segédképernyő az utasítások táblázatával átíródik a magyar nyelvű változatra.

Vizi Pál Gábor

Ahol anyanyelvi BASIC futhat

A hardveremulátoros megoldások:

Enterprise + ZX-emulátorkártya + magnó

Videoton TV-Computer + ZX-emulátorkártya + magnó

A szoftverszimulátoros megoldások:

Enterprise + ZX szimulátor pg + magnó

Enterprise + ZX, IF1, szimulátor + magnó, EXDOS-bővítő és floppy. A floppyt a BASIC microdrive utasításokkal éri el.

Videoton + ZX szimulátorprogram + magnó. A ZX szimulátor floppyról is betölthető.

Commodore 64k vagy 128k + ZX szimulátor + dataset, floppyegység. A floppyt a BASIC microdrive utasításokkal éri el.

IBM PC/AT-kompatibilis vagy azzá tehető (Amiga) gépek + ZX-Z80 szimulátor + a magnókazetta fájlváltozata mágneslemezes háttértárolón.

Spectrum-BASIC-emulátor a TV-Computeren

A TVC piacra kerülése után felvetődött az ötlet, hogy érdemes lenne más programozási nyelvet is megvalósítani. Az átírás munkaigényes, inkább a futtatásához szükséges környezet lett kialakítva a TVC-n. A különböző Z80-ra írt BASIC-interpreterek közül elterjedtsége miatt a ZX Spectrum ROM átírása volt a legcélszerűbb. A TVC-n alkalmazott memórialapozás lehetővé tette, hogy a Spectrum számára szükséges 64 K RAM, valamint a TVC ROM-ja és képernyőmemóriája megférjen. A hangszóró- és magnókezelés átírása könnyű volt, porthivatkozásokat és időzítéseket kellett kicserélni. A billentyűzetkezelő rutin átírása a konverziós logikával egyszerűen oldódott meg. A legfőbb gondot a képernyőkezelés jelentette.

A ZX képernyőjén minden karakter előtér- és háttérszínű. E színeket, valamint az adott karakter villogtatását és emelt fényerejét egy attribútumbájt határozza meg. A TVC azonos felbontással minden pontot négy színben jelenít meg, 16-os palettából. A kompromisszum: az elő- és háttérszín nem változtatható, ezt egy-egy palettaszínhez kellett hozzárendelni, a villogást a másik két palettaszín jeleníti meg a palettaértékek cserélgetése mellett. A villogó karakterekre a kurzorkijelölés miatt van szükség. Kevesebb szín mellett a képernyő információtartalma csak kevésbé csökkent. A ZX rutinok időnként írnának a ROM címekre, itt a TVC RAM-jába sikerült... A változtatásokból látszik, a ZX-BASIC szimulátoron csak a hardvert ROM-rutinokon át kezelő programok futtathatók, ilyenek elsősorban a felhasználói programok, természetesen az eredeti, majd megmagyarított ZX BASIC is. A kazettáról betölthető Spectrum BASIC alapján készült a szinte tökéletes emulátorkártya.

Bata László

CAD és CAM az üzemekben

A kivitelezés két pillére

A mérnöki tevékenység célja, hogy a gyártmányokra vonatkozó ötleteket hatékonyan működő és gazdaságosan üzemeltethető termékekké alakítsa át. A CAD feladata a termék geometriájával szorosan összefüggő információk előállítása. Ezek az információk alapul szolgálnak a termékek elkészítésének tervezéséhez, illetve a kivitelezés tényleges végrehajtásának irányításához. A CAD-adatokat tehát a különféle elemzések mellett a gyártással összefüggő feladatok számítógépes megoldása is igényli.

Amikor a konstrukciós tervezés befejeződött, a következő tevékenységeket kell végrehajtani: (a) Alkatrész-azonosítás és -osztályozás, (b) Folyamattervezés, (c) Szerszámtervezés és -készítés, (d) Anyagszükséglet-tervezés és beszerzés, (e) Számjegyvezérlésű szerszámgépek alkatrészprogramjainak előállítása, (f) Termelésütemezés, (g) Robotizált eszközök programozása, (h) Gyártás és szerelés, (i) Folyamatközi és végellenőrzés.

A fenti tevékenységek mindegyikének részben vagy egészben alapjául szolgál a CAD-adatbázis. Számítógépes megvalósításukat a fejlett ipari országokban általánosan elfogadott CIM (számítógéppel integrált termelés) koncepcióból adódó, átfogó integrálási törekvések határozzák meg. Napjainkra a CAD-en alapuló, kivitelezést támogató módszereknek és eszközöknek rendkívül széles választéka alakult ki. Az alábbiakban ezek közül, mint rövid távon is számításba veendőket, a számítógépes folyamattervezés és a minőségbiztosítás módszereit érintjük.

Híd a gyártás oldalára

Gyakorlati szempontból közelítve, a vállalati gyártási tevékenység számítógépes támogatásának két pillére a CAD és a CAM. E két pillér azonban távol áll egymástól, gyakran olyannyira messze, hogy az információ nem is jut át az egyik helyről a másikra. Ezt csak azért kívántam megjegyezni, mert a CAD és a CAM egymáshoz rendelése nem annyira kézenfekvő, mint ahogy azt a CAD/CAM betűszó alapján gondolhatnánk. A CAD-adatbázis, mint azt korábban tárgyaltuk, a geometriát leíró vagy azzal szoros kapcsolatban álló adatokat tartalmaz. Ilyen jellegű információkra a gyártás végrehajtásához is szükség van, de a CAM oldalán meghatározóak a kivitelezés eszközeire, folyamataira, paramétereire és logisztikájára vonatkozó adatok.

A tervezés és a gyártás szempontjából egyaránt fontos az alkatrészek azonosítása. Ennek során a funkciókat és a potenciális szerkezeti kapcsolódásokat mutató kódokat rendeljük a tervekben szereplő alkatrészekhez. Az azonosításra — számítógépek alkalmazása esetén — különösen hangsúlyt kell fektetni. A számítógépes rendszerekben, ha valami nem egyértelműen azonosítható, akkor attól fájdalmas búcsút kell vennünk.

Az alkatrész-osztályozás a hasonló geometriájú vagy a hasonló megmunkálásigényű alkatrészeket rendezi osztályokba. Nyugaton többféle kódolási sémát alkalmaznak. Ilyen például az OPITZ, a KK-3, a MICLASS, a CODE és a DCLASS. Az osztályozás lehetővé teszi, hogy valamely alkatrész gyártását egy kész folyamatterv felhasználásával vagy módosításával valósítsák meg.

A folyamattervezés a konstrukciós tervezési adatokat alakítja át gyártási és megmunkálási adatokká, ennélfogva a folyamattervek lényegében az alkatrészek megvalósításának és szerelésének stratégiáját adják meg. A folyamattervezés az adatigényessége miatt a hagyományos gyártás mostohahagyermeke volt. A CIM-koncepcióban viszont kitüntető szerep jut a számítógéppel segített folyamattervezésnek (CAPP), mert az összeköttetést teremti meg a CAD és a CAM között.

A korszerű folyamattervező rendszerek kialakítását elősegítette a testmodellezés, de még inkább az alaksajátosságokkal való modellezés módszertanának kidolgozása. Mint összekötő a tervezőiroda és a gyártóműhely között, a folyamattervezés jelentős kihatással van a termelékenységre és a gyártási költségekre. Mivel a vállalat egészére vonatkozó információk elengedhetetlenek a megbízható döntéshozatalhoz, az igényes CAPP rendszerek általában folyamatosan aktualizált gyármodell alapján dolgoznak. A gyártási tevékenységet több szinten optimalizálják. Régebben sorrendtervben, művelettervben és műveletelem-tervben gondolkodtak, viszont ma már finomabb megközelítések is léteznek.

Előre és vissza, de „folyamatosan"

A folyamatterv származtatásának egyik lehetséges módja, amikor azt meglévő terv módosításával alakítják ki. A másik, amelyet sokan a folyamattervezés számítógépes automatizálásának tekintenek, a generatív elvű feldolgozás. Ennek két megvalósítási formája létezik, az előrehaladó és a visszahaladó. Az előbbi a nyers darabból indul ki, és addig végzi az anyagleválasztást, amíg a kész darabot el nem éri. Az utóbbi a készből indul ki, és folyamatos pótlékolással határozza meg a kiindulási alakot. Az előrehaladó tervezés természetesebb, viszont a soron következő műveletek feltételeit a megelőzőek befolyásolják, ami automatikus végrehajtás esetén nehezen programozható. A

visszahaladó tervezés pontosan ezt a nehézséget oldja fel, de a feladatmegoldás sikerét az alaksajátosságok előre nem látható egymásra hatásai jelentősen zavarják. Következésképpen igazán hatékony visszahaladó rendszer megvalósítása csak szakértőrendszer-környezettől remélhető.

A CAPP rendszerek fejlesztésére számottevő hatást tett az a felismerés, hogy a folyamattervek tipikus műveleti tervrészekből is szintetizálhatók. Ez a félig generatív megközelítés részben feloldja azt az ellentmondást, ami a nem tipizálható geometriájú alkatrészeknek ismert műveletelemekkel való lefedésében jelentkezik. A heurisztikus tudásfeldolgozással támogatott CAPP rendszerek problémamegoldó képessége szélesebb körű, ugyanakkor fejlesztésük költségesebb és időigényesebb. Az algoritmikus programozású rendszerek hatáskörükben korlátozottabbak, viszont megvalósításuk egyszerűbb. A folyamattervezés eredménye az érintett szerszámgépekre vagy egyéb gyártóeszközökre vonatkozó gyártási utasítások listája, amit az integráltság magasabb szintjén a számítógépek közvetlenül adnak át a berendezések memóriájába. Ezeket az információs csomagokat alkatrészprogramoknak nevezzük.

NC, CNC, DNC, AC

A negyvenes évek közepén kezdeményezett amerikai kutatások eredményeként napjainkra az alkatrész-megmunkálás programozása a számítógéppel segített gyártás egyik leginkább kidolgozott területe. A lyukszalagos és -kártyás gépvezérlés (NC) a beépített számítógépek megjelenésének eredményeként számítógépes számjegyes vezérléssé (CNC) alakult át, amely már nemcsak a megmunkálási mellékidők, hanem a főidők optimalizálását is lehetővé teszi. A programozáshoz az APT nyelvet vagy származékait (például ADAPT, EXAPT, MINIAPT, UNIAPT), illetve egyéb nyelveket (például PROMPT, SPLIT) alkalmaznak. A magasabb szervezettségű gyártóegységek (például cellák, sorok stb.) működésének vezérlése központi számítógépet igényel, amely kommunikációs hálózaton keresztül juttatja el a vezérlőinformációkat a felügyelete alatt álló berendezésekhez.

Ez az ún. közvetlen számjegyes vezérlés (DNC), amely az egységbe foglaláson keresztül megteremti a lehetőségét az adaptív gyártásirányításnak (AC). A DNC a megmunkálási folyamatban végzett érzékelés alapján megváltoztatja a működési paramétereket, és így fokozza a termelékenységet és a megbízhatóságot. A megmunkálóegységbe foglalt valamennyi berendezés a visszacsatolásos szabályozás alapján működik.

Minőség körben és karikában

A minőség a fejlett országokban a társadalmi érvényesülés egyik kulcskérdése. A minőséghez kapcsolódó elvárások kiterjednek minden mesterségesen létrehozott dologra. Gyakran figyelmen kívül hagyott igazság, hogy az ipari tevékenység során létrehozott termékek minőségét legnagyobb mértékben a tervezői tevékenység határozza meg. Úgy is lehet fogalmazni, hogy a konstrukciós tervezés minőségfejlesztő, minden más termékelőállítási tevékenység minőségmegtartó szerepet játszik. Rosszul átgondolt és szegényes tervek alapján a mai technológia legmagasabb csúcsát képviselő AC-gyártás sem tud minőségi termékeket előállítani.

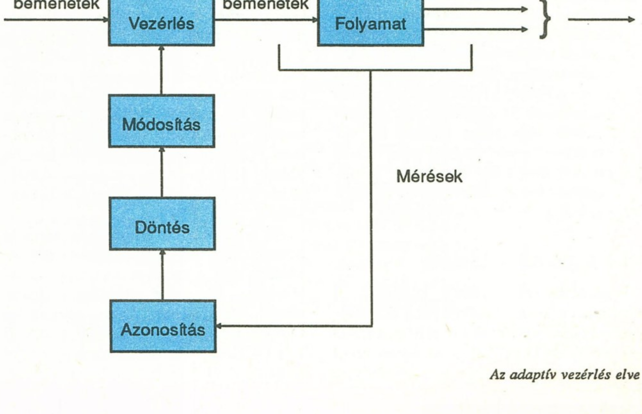
Mindamellett természetesen nem elhanyagolható az a szerep sem, amit a termékek anyagi megvalósításához kapcsolódó tevékenységek játszanak. E felismerések a hetvenes évek közepén kaptak lényegi hangsúlyt, amikor a fogyasztói piac megszerzésének és megtartásának elsődleges startégiájává váltak. Az ún. minőségi körök kialakításának hátterében az ált, hogy a termelési folyamat valamennyi elemében biztosítsák a kifogástalan minőséget, ami aztán önmagáért beszél bármely más termékkel való összehasonlításban.

De mi köze mindennek a számítógépes műszaki technológiákhoz? Csupán annyi, hogy meglátásom szerint éppen a minőségre való törekvés ösztönözte az e sorozatunk célterületét is adó CAD és az egyéb CAXX-technológiák fejlesztését, illetve széles körű alkalmazását, majd ez vezetett el a totális gyártás koncepciójának kialakulásához.

A korszerű termékek az eredeti ötletek feltárásán, több koncepcionális változat kidolgozásán és a korábbi gyártmányokra vonatkozó tapasztalatok hasznosításán alapuló tervezéssel hozhatók létre. Igazán megbízható akkor lesz a konstrukciós tervező munkája, ha az alkatrész vagy szerkezet alakjára vonatkozó elképzelését (rajzát vagy modelljét) numerikus módszerekkel is megvizsgálja. Korábbi áttekintésünk rámutatott, hogy a tervezők sokféle számítógép-orientált elemzési, optimalizálási és szimulációs technika közül választhatnak.

A módszerek mindegyike feltételezi az objektum geometriájára vonatkozó adatok könnyű módosíthatóságát. Ennek szükségességét az indokolja, hogy (a) az alakot leíró geometriai adatállományt általában — de a végeselemes analízis végrehajtásához minden esetben — egy, az elemzést lehetővé tevő geometriai állománnyá kell átalakítani; (b) a leghatékonyabb feldolgozás akkor biztosítható, ha az elemzési eredmények alapján igényelt változtatást a modellen közvetlenül (újragenerálás nélkül) lehet átvezetni. Nem kell túlságosan hangsúlyozni, hogy a hagyományos tervezői rajzok borotvapengés kapirgálásánál a CAD rendszerek minőségileg más lehetőségeket nyújtanak...

A tervezőknek meg kell győződniük az általuk elképzelt szerkezetek helyes



Az adaptív vezérlés elve

működéséről. Vizsgálódásaiknak a folyamat szimulációján túl ki kell terjedniük a szerelhetőség feltételeire is. E tekintetben valamennyi számítógéppel segített rajzolórendszer, továbbá a legtöbb CAD modellező is kívánnivalót hagy maga után. Egyszerűen azért, mert alkatrész-orientáltak. Viszont a kivitelezést követően az alkatrészek egymással összeszerelve funkcionálnak. A tervezés folyamán a szerelhetőség javításának egyik eredeti megoldása az alkatrészek számának csökkentése. Bizonyos konstrukciós kialakításokkal (például éllekerekítéssel, kúposítással stb.) a szerelhetőség ugyancsak javítható. Ez utóbbiak jelentéktelen apróságnak tűnhetnek, viszont hatásuk számottevő.

A minőséghez kapcsolódó egyéb feladatokat a minőségbiztosítás vagy a minőség-karbantartás fogalommal jelöljük. A minőségbiztosítás tartalma túlmegy a korábban általános minőségellenőrzésen, amelyet manuális mérési módszerek és statisztikai mintavételi eljárások alkalmazásával hajtottak végre. A hagyományos mérés azonban időigényes, fegyelmet igénylő, ugyanakkor monoton jellegű és hibalehetőséget hordozó tevékenység. Mindezen okok miatt a számítógépes támogatás nemcsak hogy kívánatos, de elengedhetetlen is. Meg kellett oldani a számítógéppel vezérelt mérőberendezéseknek a gyártási folyamatba való integrálását is, hogy azok ne okozzanak szűk keresztmetszetet. A sorozatmérésekre — és nem pusztán mintavételezésre — kialakított berendezések esetében számítástechnikai oldalról a nem érintkező észlelőkről származó jelek valós idejű feldolgozását kell megoldani, ami gyakran több, együtt dolgozó processzort igényel.

A számítógéppel támogatott minőségbiztosítás (CAQC) két részterületre oszlik. Egyik a folyamatközi ellenőrzés (CAI), ami például a különböző megmunkálási műveletek esetében a tűrésezett geometriai modell adatait hasonlítja össze az elkészített munkadarab ellenőrizendő méreteivel, és ennek alapján ad értékelést. A totális minőség elve megköveteli a 100 százalékos folyamatközi ellenőrzést, ami sorozatgyártásnál nagy kihívást jelent a mérési technológiával szemben. Azonban a megbízható mérés semmit sem ér, ha nincs közvetlen visszacsatolás a gyártóeszközökhöz, amelyek mondjuk szerszámkopás következtében méret feletti munkadarabokat ontanak.

Minden jó, ha a vége jó!

A munkafolyamatbeli észlelést a mesterséges látórendszerek támogatják. Ezek alkalmazásából olyan előnyök adódnak, amelyek mind a termék minőségében, mind az ellenőrzési folyamat hatékonyságában megjelennek. Napjaink látórendszerei általában 300X300 képpont valós idejű feldolgozására képesek, ha megfelelő kapacitású számítógép áll a háttérben. E rendszerek a számítógépes grafika és geometriai modellezés, illetve az elektronikus alakfelismerés és képfeldolgozás szerves egységét valósítják meg.

Még bonyolultabb feladatokat jelent a számítógép-alkalmazásban a végtermék-ellenőrzés (CAT), ugyanis sok esetben nem pusztán mérőszámok, hanem kvalitatív jellemzők és esztétikai szempontok alapján kell állást foglalni a minőségről. A probléma lényegét ez esetben az adja, hogy a CAD- vagy esetleg a CAM-adatbázisból nem nyerhetők ki azok az információk, amelyek a mérési tervek kidolgozásához szükségesek. A mérések többsége nem geometriai jellegű, bár bizonyos koordinátaméréses ellenőrzésekre általában szükség van.

A számítógépes automatizálásnak több szintje valósítható meg a CAT-ben. Alapszinten a számítógépek egyszerűen adatokat szolgáltatnak az objektum CAD-modelljére vonatkozóan, nyomon követik a tesztelést és elemzik az eredményeket. Közbenső szinten a számítógépek olyan mérési programok megvalósítását vezérlik, amelyek nagyszámú paraméter egymástól függő kiértékelésének végrehajtására képesek. A magas szintű CAT a végtermék-ellenőrzés preventív jellegű végrehajtását jelenti, ami a CAM, a CAI és a CAT teljes összefonódásaként valósul meg.

Mivel mindehhez az alapadatokat a CAD-tevékenység szolgáltatja, a kör bezárul. Ami körvonalazódott, az maga a CIM-technológia.

Horváth Imre

A Framework-világ

Saját szakálladra...

A Fred nyelv legalapvetőbb jellemzőit már megismerhették az előzőkben. Ez alkalommal olyan különlegességek kerülnek terítékre, mint a saját függvények készítése (ha valaki kevesellné azt a mintegy 170 beépített függvényt, amelyeket a Framework programozási nyelve kínál, vagy nem találja meg közöttük azokat, amelyeket nap mint nap használnia kellene), vagy a hibakezeléshez elengedhetetlenül fontos speciális függvények. (A mintaprogramokat továbbra sem kell begépelnie, a lemezmelléklet \FRED könyvtárában megtalálhatók.)

A felhasználó által definiált függvények

A FRED nyelvű programokban a formulák közötti értékátadásra két lehetőség van: részint globális változók útján, részint a függvényekhez hasonló módon (paraméterként megadva az értékeket). Az első esetre láthattak már példákat a korábbiakban. Most nézzük a másodikat!

A beépített függvények használatát ismerjük: a függvény a neki megfelelő funkciót az argumentumlistán megadott adatokkal végzi, s eredményként egy függvényértéket szolgáltat. Egy program számára a feldolgozandó adatokat billentyűzeten írtuk be, s a feldolgozást elvégezve eredményként a program utolsó kifejezésének értékét kapjuk. (Most nincs értelme szólni a paraméter nélküli függvényekről, hiszen ezeknek megfelelnek a szokásos programhívások.)

Ha egy programot függvényként szeretnénk meghívni, akkor — az adatokat argumentumokként megadva a számára — ún. **felhasználói függvényt** kell készítenünk. Ez egy olyan program, amelyben a bemenőadatokra speciális változókkal, ún. argumentumframekkel hivatkozhatunk. Ezek a változók: **@item1, @item2,...,@item16**, azaz egy saját függvény legfeljebb 16 argumentummal rendelkezhet. E változók **@item(1), @item(2), ..., @item(16)** formában is használhatók. Az argumentumframeket mint változókat deklarálni nem kell; értéküket a függvény hívásakor a paraméterlistáján megadott adatokból nyerik: a listán elsőként szereplő adat értékét @item1 (vagy @item(1)) veszi fel, a második argumentum értékét @item2 (vagy @item(2)) stb. A saját függvény értéke a függvényt definiáló program utolsó kifejezésének az értéke.

A következő programban egy olyan összeadó függvényt definiálunk, amely az első paraméterben adott értéktől a második paraméterben adott értékig összeadja a páros számokat.

```
;paros
@local(szam,osszeg),
@if(@item1>@item2,
@list(
@eraseprompt,
@beep,
@prompt("Rossz a paraméterek
sorrendje: " &
"a végérték kisebb, mint a
kezdőérték!",10)
),
@list(
@if(@mod(@item1,2)<>0,
 szam:=@item1+1),
@while( szam<=@item2,
 ossze:=ossze+szam,
 szam:=szam+2)
)),
paros:=ossze
```

Ebben a programban az @item1 argumentumframe képviseli a kezdőértéket, @item2 pedig a végértéket. Mivel ezek aktuális értéküket a fenti programmal definiált PAROS nevű függvényünk hívásakor, annak paraméterlistájáról kapják, természetesen nem állhatnak az értékadó kifejezés bal oldalán. A függvény hívása lehet például a következő:

```
@paros(35,100)
```

aminek értéke a 35 mint kezdőérték, és 100 mint végérték közötti páros számok összege. A kiszámításhoz a PAROS nevű keretre írt program kerül végrehajtásra úgy, hogy @item1 felveszi a 35 értéket, s (kukac!)item2 a 100 értéket.

A Framework Libraryjében két függvénydefiníciót találunk a **User-defined functions** szekcióban WEEKDAY és HOUR néven. Az első a paraméterként közölt dátumhoz megadja, hogy a hét melyik napjára esik. A függvény paraméter nélkül is működik, ekkor az aktuális gépi időhöz adja meg a napot. Példánk ennek magyarított, kissé módosított változata:

```
;hetinap
@local(datum,n,m),
datum:=@if(@itemcount>0,
@if(@itemcount=3,
@date(@item1,@item2,@item3),
@list(@eraseprompt,
@prompt("Három paraméter
szükséges: év,hó,nap (számmal)",20),
@return("adathiba"))),
@today),
n:=@mod(@diffdate(datum,@date(1991
,05,20)),7),
@if(n>=0,m:=n+1,m:=7-@abs(n+1)),
hetinap:=@select(m, "hétfő", "kedd",
"szerda", "csütörtök", "péntek",
"szombat", "vasárnap")
```

Függvényünk nem dátum típusú adatot vár paraméterként, hanem három számot: sorrendben az évszámot, a hónapszámot és a napot. Ha a paraméterek száma nem 0 vagy 3, hibaüzenetet kapunk, a végrehajtás befejeződik az „adathiba" szöveggel.

A felhasználó által definiált függvény hívásakor megadott paraméterek száma az **@itemcount** függvénnyel vizsgálható. A példánkban, ha nincs paraméter, a függvény a gépi dátummal dolgozik.

A saját függvények alkalmazása az interaktív feladatmegoldásoknál gyakori: táblázat vagy adatbázis adathelyeinek feltöltéséhez a megfelelő helyre beírjuk a függvényt (a beépített függvényekkel azonos módon).

Ahhoz, hogy saját függvényeinket ugyanúgy használhassuk, mint a beépített függvényeket, mindig elérhető helyen kell őket tárolni: a Library keretben. Az elkészített függvénydefiníciót tartalmazó keretet másoljuk a Library keret User-defined functions szekciójába, majd a Library keretet mentsük el!

Adatellenőrzés, programhiba-keresés

Programjainkban eddig általában feltételeztük, hogy a billentyűzeten érkeznek adatok, s azok megfelelőek. Ha azonban valamely számolós példánkban nem számot írtunk be, hibaüzenetet kapunk, és a program abortál. Hasonló a helyzet akkor is, amikor egy előkészített táblázat feltöltésénél tévedünk: oda írunk szöveget, ahová a képletterületre írt kifejezés számot vár. Szét kell választanunk a két esetet, amely egy felhasználói rendszernél szoros összefüggésben lehet.

A Framework III menürendszerében megjelent egy almenü (**Numbers** menü **Entry Format for Data**), amely lehetővé teszi, hogy az egyes adathelyek típusát még az adatbevitel előtt megadjuk. Ezekre a helyekre csak a típusnak megfelelő, értelmes adat vihető be. Ellenkező esetben sípjelet, hibaüzenetet kapunk. Ezt a lehetőséget a programjaink készítésénél figyelembe vehetjük, mert ezáltal programmal sem tudjuk hibás adatokkal feltölteni az előkészített adathelyeket. A táblázatokban, adatbázisokban levő adatok típusát programmal, különleges logikai függvények segítségével ellenőrizhetjük. Ezek: @isalpha, @isnumeric.

Hibás adatok miatt a programunk is működésképtelen lesz. Ez azt jelenti, hogy valamely kifejezés (függvény) eredménye a várt adat helyett **hibakonstans** lesz, mellé még sípjelet, hibaüzenetet is kapunk. Mivel a beépített függvényeknek egy része helyes végrehajtás esetén is egy FRED-konstanst ad eredményül, a program utolsó kifejezésétől függően előfordul, hogy a program eredménye is egy ilyen konstans. Persze az utóbbi esetben a sípjel és a hibaüzenet elmarad. Nem árt azonban utánanéznünk, hogy a függvények mit adnak eredményül.

Adathiba miatt esetleg hibássá váló kifejezések vizsgálatára a következő logikai függvények szolgálnak: @isna, @isabend, @iserr. Segítségükkel kezelhetők is az ilyen hibák. Ha például a billentyűzetről nem érkezik adat (csak az entert nyomták le), újra kell azt kérni:

```
;adat
adat:=@inputline("Adat?"),
@if(@isna(adat),
@list(@eraseprompt,
@prompt("Nem adtál adatot!
Várj!",30),
@nextkey(2),
@adat)
adat)
```

E modulban a végül is közölt adatot a programot tartalmazó keret mint globális változó adja át a főprogram számára. Az **@isna** függvény azt vizsgálja, hogy a paraméterként kapott kifejezés értéke **#N/A!** konstans-e. (Egy üres adathelynek is ez lenne az értéke, ha hivatkoznánk rá.)

A **@iserr** a következő hibakonstansok esetén ad #TRUE eredményt: #DIV/0!, #NAME?, #N/A!, #NULL!, #NUM!, #REF!, #TBD! és #VALUE!.

Az alábbi példaprogram addig adja össze a billentyűzeten beírt számokat, amíg ESC-t le nem ütünk. A @value konvertálófüggvény értéke #NAME?, ha nem számot kap a billentyűzetről, s ilyenkor a program leáll. Ezt az @iserr függvénnyel kerüljük el:

```
;ciklus2
@local(osszeg,szam),
ciklus2:=" ",
@while(@key<>{ESC},
@if(@iserr(szam:=@value(
@inputline("Írj be egy számot vagy
üss ESC-t!"))),
0,
osszeg:=osszeg+szam
)),
osszeg
```

Hibaüzenet, nyomkövetés, javítás

A FRED programok javítását egyszerűen elvégezhetjük, ha az abortált program keretszegélyére állunk, s lenyomjuk az F2-t. A szerkesztősorban a hibás programsor jelenik meg, benne inverzen a feltételezett hibás rész. A javítást az üzenetsorban egyidejűleg látható hibaüzenet is segíti. Bizonyos esetekben még ez sem elég. Ilyenkor fordulunk a Framework nyomkövetőjéhez, a **@trace** függvényhez. A vizsgálni kívánt program vagy programrész elejére írjuk be a

```
@trace(naplo,#on)
```

sort, s a végére a

```
@trace(#OFF)
```

sort. Ez utóbbi el is maradhat, ha a program végéig kérjük a naplózást. Hozzunk létre továbbá egy üres szövegkeretet, amelynek neve **naplo** legyen (ezt adtuk meg a @trace paramétereként). Ha ezután elindítjuk a programot, ez a szövegkeretben részletes naplót készít a program futásáról. Az utóbbi példaprogramunkról az alábbit:

```
TRACE Result: #ON
LOCAL
LOCAL Result: 0
SET
SET Result: " "
WHILE
 KEY
 KEY Result: {F5}
 IF
  ISERR
   SET
    INPUTLINE
    INPUTLINE Result: "5"
    VALUE
     inputline Called
     inputline Result: 5
    VALUE Result: 5
   SET Result: 5
  ISERR Result: #FALSE
  SET
  SET Result: 5
 IF Result: 5
 KEY
 KEY Result: {RETURN}
 IF
  ISERR
   SET
    INPUTLINE
    INPUTLINE Result: "9"
    VALUE
     inputline Called
     inputline Result: 9
    VALUE Result: 9
   SET Result: 9
  ISERR Result: #FALSE
  SET
  SET Result: 14
 IF Result: 14
 KEY
 KEY Result: {RETURN}
 IF
  ISERR
   SET
    INPUTLINE
    INPUTLINE Result: "j"
    VALUE
     inputline Called
     inputline Result: #NAME?
    VALUE Result: #NAME?
```

```
ISERR Result: #TRUE
IF Result: 0
KEY
KEY Result: {RETURN}
IF
ISERR
 SET
  INPUTLINE
  INPUTLINE Result:
  VALUE
  VALUE Result: #VALUE!
 ISERR Result: #TRUE
IF Result: 0
KEY
KEY Result: {ESC}
WHILE Result: #FALSE
```

A naplókeret kinyomtatható; ha már nem szükséges, törölhető.

A következő, befejező részben a tartománykezelő függvények használatáról, valamint a menükészítésről lesz szó.

Gyakorló feladatok:

1. Készítsen felhasználó által definiált függvényt, amely a magyar helyesírási szabályoknak megfelelő dátumstringet állít elő: 1991. július 16. A függvény neve DATUM. Paramétere dátum típusú, amelyet ha nem adunk meg, a definiáló program az aktuális gépi dátummal dolgozzon.

2. Írjon olyan programot, amely megadja a billentyűzeten beírt név hosszát! Ne fogadja el a program, ha nincs adat, továbbá azt se, ha szám került a betűk közé. A képernyő aljára kiírt üzenetben ne csak egy szám (a név hossza), hanem a név is szerepeljen!

Az előző szám gyakorló feladatainak megoldása:

1.

```
;jelszo jelszo.p:=0, @jelszo.jel

;jelszo.jel
@local(betu,titok,vege),
titok:="",
@eraseprompt,
@prompt("Add meg a password-öt, a
végén üss {ESC}-t!"),
@nextkey,
@while(@key{ESC},
@beep,
betu:=@keyname(@key),
titok:=titok&betu,
@nextkey),
@eraseprompt,
@if(titok<>"rejtjel",
@list(@prompt(titok & " rossz!",30),
vege:=#false,
@nextkey(2),
@eraseprompt,
@if(p>1,
@return(vege),
;@pk("{ctrl-d}qy"),
@jelszo.jel)),
@list(@prompt("A program
folytatódik",20),
vege:=#true,
;ide jön a programunk
@nextkey(2),
@eraseprompt,
@prompt("Itt fut a program!",30)
))
```

A programban nem használhatjuk az @inputline függvényt, mert a beütött karakterek megjelennek a képernyőn. Ezért karakterenként kell bevinni, majd összeállítani a jelszót. A Frameworkből való kilépéshez a programsort óvatosságból megjegyzéssorba tettük. Helyette egy @return az aktív kifejezés, amely csak a programból lép ki.

2. A program szükséges moduljai a SZAMLA konténerkeretben vannak. Innen indítjuk a programot. A számla készítése F3 leütésére indul, ez végrehajtja az UJSZAMLA programot. Ez létrehozza a számlakeretet. A beírt adatot javítani csak az enter leütése előtt lehet. Az összegzés F4-re történik, befejeződik a számla készítése, mentjük a számlát (a SZAMLA program indítása előtt beállított könyvtárba), kinyomtatjuk, majd a képernyőről töröljük. A billentyűzet szűréséhez felhasznált, már ismert formulákat itt nem írjuk le újra. A konténerkeretben levő keretek „tartalomjegyzéke":

szamla1
maszk
kilép
ujszamla
osszeg
szamkiv
programot tartalmaznak,
a globálisváltozó: mai

A **blokk** keret a számla-mintakeret. Ebből készül az aktuális számla.

```
;szamla
@szamla.szamla1

;szamla.szamla1
@keyfilter({all},maszk),
@keyfilter({esc},kilép),
@keyfilter({char},p),;szamkiv),
@keyfilter({return}),
@keyfilter({backspace}),
@keyfilter({f3},ujszamla),
@keyfilter({f4},osszeg),
@eraseprompt,
@prompt("F3 — új számla, F4 —
mentés, nyomtatás" ,15)

;szamla.szamkiv
@if(@and(@key={0},@key<={9}),
  @performkeys(@keyname(@key)),
    @maszk)

;szamla.ujszamla
@eraseprompt,
@prompt("F3-új számla, F4-mentés,
nyomtatás",15),
@nextkey(2),
mai:=@time4(@today),
@setselection("szamla.blokk"),
@echo(#off),
@pk("{f8}{ctrl-out}{return} "&mai&
"{del}{del}{del}{del}{del}{return}"),
@pk("{in} "& @date1(@today)&
"{return}{return}{return}"),
@echo(#on),
@eraseprompt,
@prompt("Minden adat végén üssön
két entert!",20)

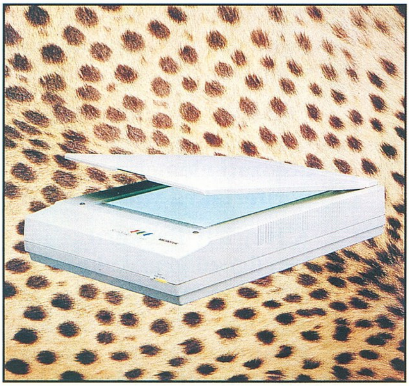
;szamla.osszeg
@local(tart,ossze),
@setselection(mai),
@pk("{F9} ------------{return}{return}"),
@pk("@row-7{return}{uparrow}{uparrow
}"),
@pk(ossze&"{return}{return}{return}"),
@pk("@sum(a1:{uparrow}{uparrow}{ret
urn})"&
"{return}{return}{return}{return}"),
@pk("viszontlátásra!{return}"),
@pk("{ctrl-return}"),
@print(mai),
@pk("{out}{del}"),
@setselection("szamla"),
@pk("{return}")
```

A program természetesen tovább is finomítható (ennek eszközeiről is lesz szó a következő részben), a billentyűzet programozásának illusztrálására most elégedjünk meg ezzel.

Kóczy A. Judit

A LETAPOGATHATÓ VALÓSÁG

SCANMAKER 600 Z/ZS



MICROTEK

Scanner · Software · Support

MICROTEK SCANNER

Felbontás: 600 DPI.

Képméret: max. 21,6 x 35,6 cm.

Letapogatás: háromszori átfutással, külön a piros, a zöld és a kék színhez.

Csatlakozás: IBM PC, PS/2 és Apple Macintosh számítógépekhez.

A Microtek magyarországi disztribútora:

Mikropo
Számítástechnikai
Kisszövetkezet
1369 Bp.
VI., Nagymező u. 51.
Telefon: 112-7830,
Fax: 112-4431

MIKROPO

INFORMÁCIÓKÉRÉS: 11 ◄



„The MACRO"

- A számítógép, mely formájában új fejezetet nyit a számítástechnikában
- A számítógép, mely minden irodában nélkülözhetetlen munkatárs
- A számítógép, melyet új bemutatótermünkben megtekinthet, kipróbálhat, letesztelhet
- A számítógép, melynek megbízhatóságára jellemző az 1 + 2 év garancia
- A számítógép, melyet üzembe helyezve, saját irodájában vehet át
- A számítógép, melyre Önnek is szüksége van!

ÚJONNAN MEGNYÍLT BEMUTATÓTERMÜNKBEN

magasabb színvonalon, kellemes légkörben, kibővült áruválasztékkal (telefonok, faxok, fénymásolók, írógépek, 3M termékek) állunk ügyfeleink rendelkezésére.

Kereskedelmi Iroda:
1016 Budapest,
Szirtes u. 28/a
Tel.: 186-5782, 186-5686,
185-7866
Fax: 186-5686 • Telex: 22-5375

Bemutatóterem/Mintabolt:
1123 Budapest,
Alkotás u. 21.
Tel./Fax: 156-4802

INFORMÁCIÓKÉRÉS: 07 ◄

UNIX-ot akarunk!

Nos, mit is szeretnénk?

Sorozatunk első része a UNIX általános jellemzőivel foglalkozott. Mielőtt rátérnénk az operációs rendszer részletes tárgyalására, most egy konkrét UNIX rendszer felépítését, megvalósításának lehetőségeit, hardver- és szoftvereszközeit ismertetjük. Mindezt egy példa segítségével próbáljuk szemléletessé tenni. Természetesen mindaz, amit itt leírunk, továbbra is a PC-s környezetre és a PC-s UNIX-okra vonatkozik.

Példánkban legyen a feladat egy pénzügyi rendszer megvalósítása. Tíz munkahelyet akarunk kialakítani, amelyből négyen majd adatrögzítés folyik, a maradék haton pedig a fejlesztők, ill. a felhasználók dolgoznak. Bizonyos adatokat grafikusan is meg kell majd jeleníteni, ezért a felhasználói munkahelyekből kettő legyen grafikus terminál. A munkahelyek több szinten helyezkednek el. Az adatrögzítők egy teremben lesznek, a többi munkahely végleges elhelyezése még nem tisztázott.

Feltétel még, hogy a rendszer legyen bővíthető, azaz legyen lehetőség a munkahelyek számának növelésére. Néhány munkahelyen DOS-os alkalmazásokat is szándékozunk futtatni. Az alkalmazói szoftverről csak annyit tudunk, hogy a Dataflex adatbázis-kezelő rendszerben íródik, és majd elkészül. (Ezt a választást egyébként a jó hazai tapasztalatok indokolják, s nem közönbös az sem, hogy ez a rendszer a DOS és a UNIX alatt ugyanúgy működik.)

A munkahelyek egy közös adatbázist használnak, amelyet egy szerver gép biztosít. A terminálokat valamilyen módon ezzel a géppel össze kell kötni. Egy ilyen rendszer kialakítására több lehetőség is van, de most csak a két legnépszerűbbet említjük, a NOVELL-t és a UNIX-ot. Bővebb magyarázat nélkül a UNIX mellett döntünk. Azt várjuk, hogy rendszerünk gyorsabb, nyitottabb, könnyebben bővíthető lesz, mintha NOVELL-t használnánk.

A feladat tehát meghatározza
— a munkahelyek számát,
— a munkahelyek jellegét (a terminálok típusát),
— a szerver gép méreteit (memória, háttértár).

Ezen adottságok mellett a szerver gép és a terminálok típusának kiválasztása, valamint ezek összekötésének megtervezése már a mi dolgunk. Első lépésként nézzük meg, hogy milyen legyen a szerver. (A továbbiakban a központi gép, UNIX gép vagy a host gép meghatározás is a szervert jelenti.)

Legfőbb a kiszolgálás

Egy UNIX-os rendszer kialakításában nagyon fontos a megfelelő szerver kiválasztása. A UNIX elvben minden megfelelő kapacitású 386-os, 486-os gépen fut, de a tapasztalat azt mutatja, hogy a névtelen távol-keleti alaplapokon bizonytalanul működik. Egy UNIX rendszer legfontosabb eleme a központi gép, ezért itt nagyon fontos a minőség. A UNIX-nál minden ezen a gépen múlik. Ha egy terminál meghibásodik, azt a többi felhasználó tudta nélkül ki lehet cserélni, de ha a szerverrel történik valami, az rossz esetben adatok, programok elvesztését is jelentheti.

Általában a UNIX-os gyártók megadják az ún. kompatibilis hardverek listáját. Ezek a géptől kezdve a diszkeken keresztül egészen a soros kártyákig olyan eszközöket tartalmaznak, amelyeken a gyártó kipróbálta és garantálja a UNIX működését. Ha lehetőség van rá, ebből a készletből kell választani. Ha nincs, akkor is célszerű egy „jobb nevű" géppel foglalkozni. Itthon az ajánlott gépek közül az ALR, a COMPAQ és a WYSE a legnépszerűbb. Ha megbízható, működő rendszert akarunk, akkor ezek közül kell vennünk.

A UNIX más kategória, mint a DOS. Itt ahhoz, hogy az operációs rendszer „érvényesüljön", legalább 4 MB memória és 40 MB diszkterület szükséges — és ezek csak alapértékek. A különböző fejlesztő rendszerek, alkalmazói programok további diszkterületet igényelnek. Több felhasználó esetén a hatékony működéshez a memória méretét is növelni kell. Esetünkben 8 MB memória és egy 80 MB-os winchester már



1. ábra. Terminálkoncentrátor

kielégítő eredményt adhat, de jobb lenne nagyobb, kb. 200 MB-os winchestert beszereznünk.

Amivel szembekerülünk...

A feladat szerint alfanumerikus és grafikus munkahelyeket akarunk kialakítani. Ehhez egyszerű soros terminálokat, grafikus X terminálokat, illetve terminált emuláló DOS PC-ket illesztünk a konfigurációba.

Az adatrögzítői munkahelyeken megfelel az egyszerű soros terminál. A hazai kínálatból a WYSE 120-as terminált választjuk, mivel jó minőségű monitora hosszabb használat során sem fárasztja a szemet. Előnye még, hogy több ASCII és ANSI terminál emulálására alkalmas.

A többi alfanumerikus munkahelyen használhatnánk szintén WYSE terminálokat, de mivel az is kikötés volt, hogy a rendszerben lehessen DOS-os alkalmazásokat is futtatni, ezekre a helyekre 286-os AT-ket telepítünk. Ezek egyenként 40 MB saját winchesterrel terminálemulációra és a DOS-os alkalmazásokhoz egyaránt alkalmasak. A nem grafikus munkahelyeken monokróm monitorokat használunk.

Hátravannak még a grafikus munkahelyek, amelyeken X terminálok vagy X terminált emuláló PC-k lehetnek. Általában az X terminál grafikus lehetőségei jobbak, de egy VGA-s AT olcsóbb, ugyanakkor DOS-os grafikus programok futtatására is alkalmas. A példa kedvéért legyen tehát az egyik grafikus munkahelyen egy X terminált emuláló VGA-s PC, a másikon pedig egy színes NCD14c típusú X terminál, ami soros vonalra és Ethernet-hálózatra is csatlakoztatható.

Összeköt(tet)ések

Lássuk ezek után, hogyan lehet a munkahelyeket összekötni a szerver géppel. Az ismert két lehetőség a soros vonal, illetve a hálózat. Nézzük meg tehát, mit kínál az egyik, és mit a másik módszer.

A soros összekötés legegyszerűbb esete, amikor a soros terminált közvetlenül a központi gép soros kártyájához kapcsoljuk. Ekkor nincs szükség külön hardverre. A szoftverrel sincs probléma, mert a UNIX tartalmazza a szükséges meghajtó programokat. Mivel ezzel a módszerrel általában két terminálnál több nem lógatható a szerverre, példánknál nem ez a megoldás.

Feladatunk megoldható lenne az ún. intelligens I/O kontrollerek segítségével. Ezek a speciális kártyák 4, 8 vagy 16 soros vonalat tudnak kezelni. Azért nevezik őket intelligens kontrollernek, mert a kártyán saját processzor és memória is található, azaz a soros portok kezelése nem terheli a UNIX-os gép központi egységét. A kártyához egy elosztó doboz csatlakozik, amelyhez soros kábeleken kapcsolhatók a terminálok, terminált emuláló PC-k vagy más soros eszközök (modem, soros nyomtató). A kártyát nem elég csak a gépbe helyezni, a kártyát kezelő programot is installálni kell, ami a UNIX megfelelő segédprogramjával könnyen elvégezhető. (Az SCO UNIX-nál például ez a custom programon keresztül történik.) A kártyákból egy gépen belül max. 4 használható, ez max. 64 terminál kiszolgálását teszi lehetővé. Hazánkban a Computone és a Chase Research cég ilyen kártyái a járatosak.

Természetesen a COM1 és COM2 portok az intelligens kontrollerek mellett ugyanúgy használhatók, ezekhez egér, terminál vagy más, soros vonalat igénylő eszköz (például modem) kapcsolható.

Általában nincs lehetőség arra, hogy négy intelligens I/O kontroller kártyát egy gépben működtessünk, vagy ha igen, gondot okozhat a központi géptől kiinduló kábelrengeteg. Esetünkben a tíz munkahelyet egy 16-os kártyával ki lehetne szolgálni, de amellett, hogy sok vezeték távozna a szervertől, és a terminálok helye kötött lenne, két másik dolog is a soros vonal ellen szól. Egyrészt a terminál nem lehet 15 méternél távolabb a szervertől, másrészt a grafikus terminálok hatékony használatához a soros vonal sebessége nem elegendő. Példaként vett rendszerünk tehát ezzel az eszközzel megvalósítható, de nem ez lesz a legjobb megoldás.

Vizsgáljuk meg ezek után a terminálok csatlakoztatásának másik lehetőségét, a hálózatot. UNIX-os rendszereknél az Ethernet-hálózatot használják, ahol a gépek a TCP/IP protokoll szerint kommunikálnak egymással. Ez a megoldás drágább, mint a soros vonal, de sok olyan tulajdonsága van, ami megéri a többletköltséget.

A hálózat sokkal gyorsabb, mint a soros vonal. Ez elsősorban a grafikus X terminálok használatánál jelent majd előnyt, de érezhető lesz valamennyi munkahelyen. A hálózat könnyen bővíthető, újabb munkahelyek és szerverek is egyszerűen bekapcsolhatók a többi mellé. A terminálok áthelyezhetők, azaz a hálózat átkonfigurálható.



2. ábra. Intelligens I/O bővítőkártyák

Feladatunkban soros terminálokat, DOS PC-ket és egy X terminált kell a hálózatra kapcsolni. Nézzük meg ennek a lehetőségeit.

Terminálok — koncentráltan

Soros termináljainkat egy ún. terminálkoncentrátoron át lehet a hálózatra kapcsolni. Ez általában 8 vagy 16 szabványos soros csatlakozási lehetőséget biztosít. A soros portokhoz kötött perifériák (terminál, modem, printer, fax stb.) a koncentrátoron keresztül, hálózaton érik el a szervert és a hálózat más gépeit. Ez az eszköz megkönnyíti a terminálok csatlakoztatását, mivel bárhol a hálózatra kapcsolható, és így nagyobb távolságok is áthidalhatók. Ezzel a megoldással a terminálok száma sincs korlátozva, mivel a hálózaton tetszőleges számú koncentrátor működhet, vagyis a bővítés lehetősége is adott. Kedvező az is, hogy a szervertől csak egy kábel megy el, és installálásakor nem kell ezt a gépet megbontani.

Ha több szerver is van a hálózaton, a soros terminálról bármelyikre be lehet jelentkezni, sőt egyes koncentrátorok azt is lehetővé teszik, hogy egyszerre több géppel legyünk kapcsolatban. Ezt biztosítja például a Chase Research IOLAN nevű terminálkoncentrátora is, amelynek fényképe a 2.ábrán látható. A terminálkoncentrátor, amint az az eddigiekből ki is derül, szintén intelligens, saját számítási kapacitással és szoftverrel rendelkező eszköz, amely a UNIX-os hálózatoknál jól alkalmazható. Mivel a példánknak választott rendszerben soros terminálokat is akarunk használni, szükségünk van egy ilyen koncentrátorra. Esetünkben az IOLAN 8 soros portot kiszolgáló változatából egy elegendő, mivel négy darab közös helyiségben elhelyezett soros terminált kell a hálózatra kapcsolni, és marad még négy bővítési lehetőségünk is. Ennél nagyobb bővítéshez újabb koncentrátorokra lesz szükség.

Terminálként konfigurált PC-ink egyszerűen a hálózatra akaszthatók. Nem kell ehhez más, csak egy csatolókártya és a kártyát kiszolgáló szoftver. Leggyakrabban a 3COM Etherlink 3c503 és a Western Digital WD8003E csatolókártyákkal találkozhatunk, ha a már megvalósított ilyen rendszereket tekintjük. Szoftverként az FTP PC/TCP programcsomagja ajánlható, mert vele a DOS-os gép a TCP/IP alapú hálózaton kommunikálhat. Ez a szoftver — a terminálemuláción túl és a fájloknak a gépek közötti másolása mellett — lehetővé teszi azt is, hogy az ún. NFS-en (Network File System) keresztül a UNIX szerver kijelölt könyvtárait logikai DOS meghajtóként használjuk. Ez azt jelenti, hogy egy DOS-os alkalmazás közvetlenül a UNIX fájljait manipulálhatja.

Az X terminálokkal sincs gondunk: ezek a szabványos TCP/IP protokollal működnek, így simán illeszkednek a hálózatba. X terminál PC-s emulációjához pedig rendelkezésre állnak a megfelelő emulátorszoftverek. Ilyen például a PC-Xview vagy a PC-XSight.

Új rendszer — régi rendszer

Rendszerünk tehát hálózat segítségével is megvalósítható. A munkahelyek jellegét, a terminálok típusát a soros megoldáshoz képest nem kell változtatni, mivel megvannak a megfelelő eszközök, amelyekkel a hálózatra kapcsolhatók. A hálózat bővíthető, újabb terminálok telepítése vagy a meglévők átrendezése nem okoz problémát. X.25 kapcsolaton keresztül lehetőség van nagy távolságú összeköttetésekre is. A példafeladat megvalósítására ezt a módszert javasoljuk. Természetesen, mint azt be is mutattuk, nem ez az egyetlen út. Vannak más tényezők is, amelyek befolyásolhatják a rendszer kialakítását, pédául a pénztárcánk tartalma.

Látható, hogy az optimális megoldáshoz ismerni kell az eszközkészletet, és szükség van az ilyen rendszerek kialakításánál szerzett tapasztalatra is. Célszerű ezért ilyen hálózatok tervezésénél szakemberekhez fordulni. Ezt azért hangsúlyozzuk, mert Magyarországon még nem ez a gyakorlat, és így egy adott feladatra nem mindig a legjobb megoldás születik.

Végül szeretnénk még két kérdéssel röviden foglalkozni. Az egyik az, hogy mit tegyünk, ha egy meglévő, például NOVELL rendszerről akarunk áttérni a UNIX-ra, a másik pedig, hogy miként lehet megtanulni a UNIX használatát.

Ha egy működő rendszerről egy másikra akarunk áttérni, a legnagyobb fejtörést mindig az okozza, hogy mi legyen a már meglévő hardverrel és a szoftvereszközökkel. Ha mondjuk egy NOVELL-hálózatot akarunk UNIX-ra cserélni, a költségeket csökkentheti, hogy felhasználjuk a már kiépített Arcnet-hálózatot és a rá csatlakozó PC-ket. Ebben az esetben az Ethernetes szoftverek nem használhatók, de a XEUS Iroda XEUS nevű, intelligens terminálrendszerével ez a feladat is megoldható. A XEUS az Arcneten keresztül biztosítja a terminálemulációt, illetve a UNIX-fájlok közvetlen elérését. A PC-k az előzőek szerint szintén felhasználhatók.

A szoftverek átvitele nehezebb kérdés. Elég sok népszerű DOS-os programnak létezik azonban már UNIX-os változata is, ami megkönnyíti a megszokott alkalmazások használatát.

Tanulni az újat

Említettük már, hogy a UNIX más kategória, mint a DOS. Ez nagyobb hardverigénye mellett szokatlan kezelésében is megnyilvánul. Nem elég egy rendszert megtervezni, megvalósítani, azt használni is kell. Különös gondot kell tehát fordítani a UNIX-felhasználók "kiképzésére" is. Ehhez ma már itthon is nagyon jó tanfolyamok állnak rendelkezésre; érdemes ezekre odafigyelni. Minden új dolog megismerése ráfordítást igényel, de sokszor nem tudjuk, hogy merre induljunk el, mit olvassunk, mit használjunk. Így van ez a UNIX-nál is, itt nagyon nagy szükség van az ilyen kapaszkodókra.

Déri Gábor

A Pascal Saga

Ne rémüljön meg, kedves olvasó, ezúttal nem egy Forsyth, Ewing vagy Guldenburg (hogy Takácsékról és Szabóékról ne is beszéljek) családról szóló, véget nem érő tv- vagy rádiósorozat forgatókönyvének első részét tartja kezében. Ez csupán a programozási nyelveket bemutató sorozatunknak a Pascal programozási nyelvről és utódairól szóló része. Folytatása persze azért ennek is lesz, de ígérjük, hogy decemberi számunkban már véget is ér a Pascal Saga története.

Az 50-es és 60-as években született programozási nyelvek (Fortran, Cobol, Algol, LISP, Basic stb.) nagy száma ékes bizonyítékként szolgált a szakembereknek a magas szintű programozási nyelvek létjogosultságára. A 60-as évek vége felé azonban egyre több kritikával illették a már meglévő nyelveket. A bírálók szerint ezekből a nyelvekből hiányoztak a program- és adatstrukturáláshoz szükséges eszközök. Hogy a bírálóknak mennyire igazuk volt, mi sem bizonyította jobban, mint a 60-as évek végének, 70-es évek elejének úgynevezett nagy szoftverválsága. A korai nyelveken íródott programok karbantartása rosszul átgondolt struktúrájuk miatt szinte lehetetlen volt, s ez sok gondot okozott. A programok továbbfejlesztése, javítása óriási erőfeszítésekbe került, az ebből fakadó financiális vonzatokról nem is beszélve.

Elhatárolt részletek

A számítástechnikai szakemberek többsége természetesen már idejében felismerte a korai nyelvek hiányosságaiból fakadó korlátokat, és az egyetemi szakemberképzésben egyre több helyen kezdték alkalmazni a strukturált programozás elvét. A strukturált programozási metodika lényege, hogy a programtervezés során az adott probléma szerkezetét módszeresen fölvázolva az elkészült program szerkezete a program által megoldandó problémákat oly módon tükrözze, hogy abban a részproblémák megoldása jól elhatárolható legyen. Ez a szempont a programok javítása, karbantartása és fejlesztése szempontjából alapvetően fontos. A strukturált programozási szemlélet megtalálható minden modern programtervezési módszerben. Fontos, hogy a strukturált programozás fogalmát ne tévesszük össze a strukturált forráskód fogalmával. Azért, mert egy program „formatervezett" — már ami a külalakját illeti —, még nem biztos, hogy strukturált. A lényeg nem a tabulátorpozíciókon múlik! Nem a külcsín, hanem a „belbecs" a fontos: egy program jó lehet, de szép nem.

1968-ban Niklaus Wirth a Zürichi Műszaki Egyetem (ETH = Eidgenössische Technische Hochschule) professzora a diákjai számára a strukturált programozási metodika elvét támogató

Névjegy

A Pascalt Niklaus Wirth, a Zürichi Műszaki Egyetem professzora tervezte az egyetemi programozás oktatási eszközének 1968-ban. A nyelvnek közvetlen elődei nincsenek, bár az Algol és az Algol-68 nyelvek hatása kétségtelenül felfedezhető a Pascalban. A Pascal magas szintű problémaorientált, strukturált programozási nyelv. Mivel a Pascal megalkotásakor Wirth elsősorban a tantermi alkalmazást tartotta szem előtt, ezért a nyelv eredeti formájában nem volt alkalmas nagy programrendszerek vagy rendszerprogramok írására. Ennek ellenére a Pascal nagyon gyorsan elterjedt akadémiai körökben, és a felsőoktatási intézmények elsődleges programozási nyelvévé vált.

1974-ben megjelent Jensen és Wirth nyelvdefiníciós könyve a „Pascal User Manual and Report", ami a standard Pascal definíciójának tekinthető. A Pascalt az évek során nagyon sokan és sokféleképpen bővítették, ami a nyelv elterjedését elősegítette ugyan, de a verziókat meglehetősen inkompatibilissá tette. A bábeli zűrzavar feloldására irányuló kísérletek (ANSI, ISO, UCSD) közül az ISO-szabvány tekinthető sikeresnek, mivel az amerikai kormányszervek és egyetemek számára előírás az ISO Pascal használata (mármint Pascal programok esetén). Wirth és kollégái a Pascal toldozgatása helyett egy új nyelvet, a Modula-2-t hozták létre, amely többek között a Pascal hiányosságait is kiküszöböli. A zürichi kutatómunka azonban nem állt meg a Modulánál.

A család legifjabb tagja, az Oberon 1988-ban született meg, és bár még nincs hivatalos definíciója, máris érezteti a hatását a legújabb Pascal- és Modula-2-implementációkon. A Pascal család közeli rokona az Ada programozási nyelv, amelyet az Amerikai Védelmi Minisztérium terveztetett, és gyakorlatilag a Pascalt mint részhalmazt tartalmazza. Az Ada elsősorban óriásrendszerek és beágyazott rendszerek programozására alkalmas, bár nagyon sokak véleménye szerint a nyelv definíciója nem elég egységes és átgondolt, sok felesleges funkciót tartalmaz, ami csak megnehezíti a fordítók írását, de gyakorlati hasznuk nincs. Az Ada egyébként nem egyszerű programozási nyelv, hanem amerikai katonai szabvány.

A standard Pascal kulcsszavai

A „Pascal User Manual and Report”-ban definiált kulcsszavak:

and	downto	if	or	then
array	else	in	packed	to
begin	end	label	procedure	type
case	file	mod	program	until
const	for	nil	record	var
div	function	not	repeat	while
do	goto	of	set	with

programozási nyelvet keresett. Mivel azonban sem a széles körben elterjedt programozási nyelvek, sem az akadémiai körökben kifejlesztett nyelvek nem nyerték meg a tetszését, munkatársaival egy új, elegáns oktatási nyelvet tervezett. Ezt a nyelvet Blaise Pascalról, a XVII. századi nagy francia matamatikusról és filozófusról nevezte el.

Bár kétségtelen az Algol 60 hatása a Pascalra, mégis a Pascal markáns jelemzői — merev programstruktúra és az adattípusok szigorú elkülönítése — teljesen Wirth szemléletmódját tükrözik. Az első Pascal-implementáció 1970-ben látta meg a napvilágot az ETH Informatikai Intézetében. Kathleen Jensen és Niklaus Wirth 1974-ben publikálta a „Pascal User Manual and Report”-ot. A könyvben leírt Pascalt nevezik standard Pascalnak. A Wirth által tervezett nyelv számos korlátozást tartalmazott. Nem volt alkalmas interaktív jellegű (párbeszédes) programok írására, magas szintű képernyő-kezelési funkciókról nem is beszélve. A fájl-, illetve be- és kivitelkezelés absztrakt módon szerepelt a nyelvben. (Tulajdonképpen a két általános célú utasítás, a read és a write segítségével zajlott minden adatmozgatás.) A nyelv nem tette lehetővé az operációsrendszer-hívásokat, sem a számítógép perifériáinak kezelését. Mivel Wirth a nyelvet kizárólag oktatási célokra tervezte, hallani sem akart arról, hogy olyan bővítésekkel lássa el, amelyek azt piaci termékké tették volna.

Kilépve az egyetemről

A Pascal azonban nem sokáig maradt az egyetemek falai mögött. Az ETH-n kifejlesztett pszeudokód Pascal-fordító segítségével a Pascal gyorsan elterjedt először más egyetemeken, majd a gyakorlati alkalmazások terén. A pszeudokód lényege a Pascal programok különböző gépek közötti egyszerű transzportálhatósága volt. A pszeudofordító gépi utasítások helyett pszeudoutasításokat (kódot) állított elő, s ez teljesen gépfüggetlen volt. Az adott célgépeken futó pszeudointerpreter vagy pszeudoassembler segítségével aztán a programot gépi kóddá lehetett átalakítani.

Mivel a pszeudofordító Pascalban íródott, és nem gépspecifikus tárgykódot állított elő, az implementáció során csak a pszeudokód—gépi kód fordítót kellett megvalósítani, ami már szinte gyerekjáték volt. A pszeudokód-fordító célja a Pascalnak a különböző nagygépeken történő gyors implementálása volt, ennek ellenére legnagyobb hatását a mikroszámítógépes világra fejtette ki. Az akkor a Kaliforniai Egyetemen (UCSD = University of California San Diego) dolgozó Ken Bowles a pszeudofordítót minden abban az időben létező processzorra és rendszerre implementálta, ezzel a programozók és érdeklődők millióit ismertetve meg a nyelvvel. A pszeudofordítóra épülő UCSD p-System operációs rendszer és fejlesztői környezet minden jelentős mikroszámítógépen futott.

Ezenfelül a nagyobb cégek saját céljaikra speciális bővítésekkel is ellátták a nyelvet. Ilyenek voltak az operációs rendszerek írásához szükséges bővítések, az irányítástechnikai és más ipari alkalmazásokhoz szükséges (real-time) bővítések, nagy programrendszerek megvalósításához külön fordítást lehetővé tevő bővítések, és még ki tudja, mi minden más célt szolgáló bővítések. A közös ezekben a nyelvekben szinte már csak a név volt, ami minden bővítés ellenére Pascal maradt.

A 80-as években több kísérlet is született a nyelv szabványosítására, azonban a fordítógyártók érdekütközései miatt ezek nemigen vezettek sikerre. A legelterjedtebb szabvány az ISO Pascal, amelynek használata Pascal-fordítók használatakor az amerikai kormányhivatalokban és oktatási intézményekben előírás.

A sokszínűség ellenére a Pascal-kód kompatibilitása semmivel sem rosszabb a többi magas szintű nyelvénél. Sőt, mivel a Pascal-forráskód jól olvasható és értelmezhető — ez a Wirth professzor által létrehozott programstruktúra érdeme —, a Pascal-dialektusok konvertálása könnyebb, mint más nyelvek esetén.

A Pascal programok felépítése

A Pascal nem tesz különbséget a kis- és nagybetűk között, ezért a kulcsszavak és azonosítók tetszőleges formában írhatóak. A programok a „program” kulcsszóval és a program nevével mint azonosítóval kezdődnek. A program törzsét megelőzi a deklarációs rész, ahol állandókat, típusokat, változókat, függvényeket és eljárásokat definiálhatunk.

```
programnév;
globális deklarációk:

const     állandó deklarációk;
type      típusdeklarációk;
var       változódeklarációk;
```

Függvények és eljárások deklarációja például:

```
function f (paraméterlista):
visszatérő érték;
lokális deklarációk
begin
utasítások
end;
procedure p1 (paraméterlista);
lokális deklarációk
begin
utasítások
end;
a programtörzs:
begin
utasítások
end.
```

A Pascal blokkorientált strukturált nyelv, és mint ilyen, rendet és olvashatóságot biztosít a megírt programok számára. A Pascalban fontos az egyes blokkok elhelyezkedése. Először a konstans-, aztán a típus-, változó-, függvény- és eljárásdeklarációk követik sorban egymást, végül pedig a program törzse következik. A program kötelezően rendelkezik névvel. A Pascalra jellemző tagolt forráskód csak tipográfiai konvenció, és csak a program jól olvashatóságát szolgálja. A fordító számára a forráskód formátuma teljesen indifferens. Egy Pascal program tetszőleges számú alprogramot, úgynevezett függvényt és eljárást tartalmazhat. Az alprogramok a Pascal programok lényeges részét képezik. A program működése tulajdonképpen az eljárások és függvények működésén alapul. Az alprogramok — akárcsak maga a program — rendelkeznek deklarációs részekkel. Az alprogramok tetszőleges mélységig egymásba ágyazhatók.

Gúzsba kötve?

Azonkívül, hogy a Pascalban a program részeinek meghatározott helye van, a Pascal nagyon kényes a különböző adattípusok használatára. Az egész számok és a karakterek teljesen különböző dolgot jelentenek, függetlenül attól, milyen az ábrázolásuk a számítógépben. Egy egész szám és egy karakter összeadásának a Pascalban nincs értelme, ezért a fordító meg sem engedi az ilyen műveleteket. Szerencsére persze vannak olyan függvények, az úgynevezett transzfer függvények, melyek segítségével a szükséges konverziók elvégezhetőek.

A Pascal összetett típusai méretüket a program futási ideje alatt nem változtathatják meg. Ha egy tömböt 10 eleműnek deklaráltunk — márpedig a deklaráció kötelező — akkor az sohasem lehet nagyobb vagy kisebb, mint a deklarált méret. Ez megakadályozza az olyan jellegű hibákat, mint amikor a memória adatterületét a program egyszerűen szétírja a pontos deklaráció hiányában.

A Pascal kritikusai gyakran nevezik a nyelv szigorú konvencióit hibáknak: szószátyár, a programozót gúzsba köti és így tovább. Mint minden az életben, ez is relatív. Ha a cél a programírásba befektetett munka megőrzése és kamatoztatása hosszú távon, akkor a Pascal szigorúsága, amit a kritikusok korlátozásnak neveznek, csupán a cél eléréséhez szükséges eszköz.

Villányi László

A Borland nyelvet nyújt

A Borland International cég Turbo Pascal-fordítója kétségkívül a legsikeresebb Pascal-fordító. Bár a legújabb verziók kizárólag IBM PC-kompatibilis gépeken futnak, az eladott fordítók számát tekintve mégis a Turbo Pascal a világ legelterjedtebb Pascal-fordítója. 1983-ban a Borland International olyan új típusú Pascal-fordítót dobott piacra, amely egy teljes editálás/fordítás ciklust kis programok esetén akár 15-20 másodpercre is redukálhatott.

Ez a figyelemre méltó termék, a Turbo Pascal, egyetlen 36 K-s csomagban egy editort, egy gyors fordítót és néhány alapvető debug funkciót tartalmazott. A fordítási-tesztelési ciklus drasztikus lerövidítése mögött egy egyszerű trükk rejtőzött. A Turbo környezet minden része — fordító, editor, forrásprogram — a memóriában volt tárolva, semmit sem kellett lemezről beolvasni. A környezet úgy dolgozott, mint a villám, és nem csoda, hogy rövid időn belül a Pascal-programozók kedvencévé vált. A gyors és kényelmes környezeten kívül a rendszerhez tartozott még sok hasznos kiterjesztés az operációs rendszer és a perifériák kezelésére. A Turbo Pascal korai verziói nemcsak MS/PC—DOS alatt működtek, hanem léteztek CP/M-80-as és CP/M-86-os változatok is.

Természetesen Borlandék az évek során egyre tökéletesebb verziókkal jelentek meg a piacon.

Az 1985-ben megjelent 3.0 verzióval a Turbo Pascal a maga 300 ezres példányszámával a legelterjedtebb Pascal-implementációvá vált. A 3.0 jelentős bővítéseket tartalmazott az eredeti verzióhoz képest, de a professzionális felhasználók igényeit nem elégítette ki. A problémák elsősorban a memóriamodellek és az elkülönített fordítás hiányában, valamint az 5000 sornál nagyobb forráskódok fordításakor tapasztalt lelassulásában nyilvánultak meg. Az 1987-es év fordulópontot jelentett a Turbo Pascal történetében. Az 1985-ben megjelentetett 3.0 verzió már megérett a cserére, a kérdés csak az volt, mire cseréljék le. Már nem sokkal a 3.0 verzió megjelenése után bejelentették, hogy készül a TurboModula fordító is. Az egyetlen „problémát" a 3.0 sikere jelentette: vajon célszerű-e két hasonló nyelvet támogatni. Bár a Modula-2 kétségtelenül felfogható szuper Pascalként is, azért mégsem várható el több tízezer programozótól, hogy nyelvet váltson. Ugyanakkor a lehetséges és igényelt Pascal-bővítések nagyon sok esetben hasonlítottak a Modula-2 megoldásaihoz.

Végül is a Borland a két projekt összeházasítását választotta, és létrejött a Turbo Pascal 4.0, ami — nyugodtan állíthatjuk — önálló nyelvvé nőtte ki magát az évek során. Az új verzió tartalmazza a Modula-2 sok funkcióját és bővítését a Pascallal szemben, bár nem az eredeti formában, sőt az 1987-ben még csak csírájában létező Oberon egyes megoldásait is. A döntés persze nem okozott osztatlan sikert a Borland fejlesztői között, akik közül sokan kiléptek a cégtől, és létrehozták a JPI céget, amelynek első terméke, persze nem véletlenül, egy Modula-2 fordító volt. Igaz ugyan, hogy a TopSpeed Modula-2 messze felülmúlta a 4.0-s Turbo Pascalt, a Borland vezetői mégis jól döntöttek, mivel a Turbo Pascal-felhasználók között az új verzió nagy sikert aratott. A nyelvi módosításokon túl, melyek közül kétségtelenül a moduláris unit struktúra a legfontosabb, a rendszer új könyvtárakat és beépített eljárásokat kapott a felhasználói igényeknek megfelelően. A Turbo Pascal kisebb-nagyobb hibáit az 1988 végén megjelent 5.0 verzióban javították ki. Az 1989-ben megjelent 5.5 verzió leglényegesebb bővítései az objektumorientált nyelvi kiterjesztések voltak, amelyek a Turbo Pascal-felhasználók széles köre előtt nyitották ki az OOP kapuit.

A tavalyi év végén megjelent 6.0 és Windows-verziók az OOP-eszközök tökéletesítését, illetve a MS Windows-alkalmazások fejlesztését támogatják új fejlesztői környezetbe csomagolva.

Apu, hogy megy be?

A Clippert nyelvi szinten bemutató sorozat előző részében a hibakezelésről és ebből kifolyólag a hibaobjektumokról volt szó. Ha már elkezdtünk objektumozgatni, akkor ne álljunk meg félúton. Ebben a részben a billentyűzeten keresztül történő adatbevitellel és ennek kapcsán többek között a GET objektumokkal fogunk foglalkozni.

Mielőtt a kiválasztott téma legnagyobb halára vetnénk rá magunkat, nézzünk körül, milyen adatbeviteli lehetőségeket kínál a Clipper. Az adatbeviteli eljárásokat három csoportba sorolhatjuk Az első csoportba a billentyűzetet karakteresen kezelő függvények tartoznak. A második a stringek bevitelét hivatott megoldani, a harmadik csoport pedig a GET objektumokkal való adatbevitelnél használatos.

Stringek konzolról való beolvasására két utasítást találhatunk. Az egyik az ACCEPT, a másik az INPUT. A két utasítás szintaxisa teljesen azonos. Mindkettőnek meg kell adnunk egy változót, amelyben a beolvasás eredményét fogjuk visszakapni. Ha a változó még nem létezik, akkor private-ként létrejön. A bevitel közben csak két szerkesztőbillentyűt használhatunk. Az egyik a backspace, amely a kurzortól balra lévő karaktert törli, a másik pedig az enter, amely befejezi a szerkesztést. A két utasítás között azonban mégis van lényegbeli különbség. Az ACCEPT a szerkesztési puffer tartalmát változtatás nélkül adja át a változónak. Az INPUT az enter lenyomása után a szerkesztési puffer tartalmát makróként értelmezve végrehajtja, és a paraméterként megadott változó ennek a kifejezésnek az eredményét fogja megkapni. Vigyázni kell, mert ha a kifejezés makróként nem értékelhető ki, akkor egykettőre a hibakezelő rutinban találjuk magunkat, ami nem feltétlen kívánatos. Bár néha egy kicsit több munkát igényel, mégis jobban járunk, ha az ACCEPT mellett maradunk.

A billentyűzetet karakteresen is kezelhetjük. Az új Clipper-verzió ezen a téren is tartalmaz újdonságokat.

Az INKEY( ) függvénynek egy numerikus paramétert adhatunk meg, amely a billentyűlenyomásra való maximális várakozás idejét írja elő másodpercekben. Amennyiben ez idő alatt nem történik billentyűlenyomás, úgy az INKEY(nullával tér vissza, egyébként a lenyomott billentyű „Clipper" kódjával. Ha paraméterként nullát adunk át, akkor korlátlan ideig változik a billentyűlenyomásra, ezzel elavulttá téve a régi verzió WAIT utasítását.

Új függvény a NEXTKEY( ), amelyet paraméter nélkül kell meghívni, és a billentyűzetpuffer következő kiolvasásra váró karakterének „Clipper" kódját adja meg, anélkül, hogy a karaktert kiolvasná a pufferból. Még két függvény tartozik ebbe a témakörbe, a LASTKEY(és a KEYBOARD(. A LASTKEY(a pufferból utoljára kiolvasott karakter kódját adja meg, a KEYBOARD(pedig az argumentumként megadott karaktert helyezi el a billentyűzetpufferben.

Az adatbevitel legjobban támogatott módja a GET objektumok használata. Ezt a módszert választva minden egyes beolvasandó adathoz létre kell hoznunk egy GET objektumot, majd ezeknek a GET objektumoknak a segítségével hajthatjuk végre az adatok beolvasását. Az ilyen módszerrel végzett adatbeolvasás áttekintése során nem lesz szó a GET és read makrók használatáról, mivel ezek csak a régebbi változatokkal való kompatibilitást szolgálják, és nem használják ki az új rendszer előnyeit.

A GET változók beolvasása közben teljesen a kezünkben marad a gép vezérlése. A billentyűzet olvasásáról is nekünk kell gondoskodni, és a beolvasott karaktertől függően vagy elhelyezzük azt a GET pufferben, vagy végrehajthatunk valami más akciót is. Ez a más akció lehet például egy helpképernyő vagy más üzenet kiírása, a GET kijelzés képernyőkoordinátáinak megváltoztatása, vagy bármi más is.

Már ebből is következik egy nagy előny az előző verziókhoz képest. Egy GET lista beolvasása közben (ha szükséges) beolvashatunk egy újabb GET listát is. A beolvasások korlátlan mélységig egymásba ágyazhatóak, anélkül, hogy zavarnák egymást.

Most nézzük meg lépésről lépésre, hogyan kell létrehozni egy egyszerű GET listát, és hogy tudjuk azt beolvasni.

Tegyük fel, hogy három olyan változónk van, amelynek az értékét a felhasználótól kell bekérni. dV1 dátum, cV2 string és nV3 numerikus. Első lépésben mindhárom változóhoz létre kell hoznunk egy-egy új GET objektumot. Ezután az egyes GET objektumok exportált változóinak kitöltése következik. Az ebben a fázisban kitöltendő objektumváltozók listáját a lemezmellékleten lévő 1. táblázat A jelű bekezdése tartalmazza. Igazából csak az első két változó kitöltése elengedhetetlen, a többinek mind van használható alapértelmezett értéke, aminek megtartása nem minden esetben jó, de fatális hibát biztosan nem okoz.

Amikor a változók beolvasására kerül a sor, akkor az INKEY(függvény segítségével be kell olvasnunk a billentyűzeten leütött karaktereket. Minden egyes karakter beolvasása után eldönthetjük, hogy mi legyen a következő lépés. A választási lehetőségek száma gyakorlatilag korlátlan. A triviális esetek a következők:

1. Legegyszerűbb esetben a beolvasott karaktert beírjuk a szerkesztési pufferbe. Ezt az INSERT(vagy az OVERSTRIKE(külső eljárás segítségével tudjuk megtenni.

2. Egy másik lehetőség, hogy az aktív GET-mezőben megváltoztatjuk a kurzor pozícióját. Például azért, mert kurzorvezérlő billentyűt nyomott le a felhasználó, vagy csak azért, mert olyan „kedvünk" van. Ha erre a döntésre jutunk, akkor a lemezmelléklet 2. táblázatának A jelű bekezdésében ismertetett külső eljárásokból választhatunk.

3. Ha egy kicsit tovább akarunk menni, akkor törölhetünk néhány karaktert a szerkesztési pufferből. Ennek a végrehajtásában a 2. táblázat B jelű bekezdésében felsorolt eljárások tudnak segítséget nyújtani.

4. Az adott helyzetben aktuális helpszöveget írhatunk a képernyőre. Ehhez nincs is szükség az objektum változóira vagy eljárásaira.

5. Megtréfálhatjuk a felhasználót, ha beolvasás közben megváltoztatjuk a GET mező képernyő-koordinátáit.

6. Az adott GET változó beolvasását befejezhetjük úgy, hogy aktualizáljuk az értékét, vagy úgy, hogy a változtatást nem jegyezzük fel. Egy befejezett GET változóhoz bármikor visszatérhetünk, a GET-ek egymásba ágyazásának semmi nem szab gátat.

A lemezmellékleten található getobj.prg program röviden tartalmazza azokat a lehetőségeket, amelyek a rendszerhez adott példaprogramokban nem szerepelnek.

Fridl György

Modula-2

Rekordok — variálva

Az előző részben megismerkedtünk a tömb adattípussal, amely lehetővé teszi, hogy egyszerre több, azonos típusú adattal könnyen dolgozzunk. Ebben a hónapban a rekordtípus kerül terítékre, amely eltérő típusú adatok együttes kezelésére ad módot.

Sok program — például adatbázis-kezelők, táblázatkezelők, katalógusprogramok vagy indexelő programok — az információt úgy generálják, hogy különböző típusú adatokat csoportosítanak össze. Hogy ilyen és más hasonló feladatokat megoldó programokat kényelmesen írhassunk, a Modula-2-ben értelmezték a rekordtípust, ami lehetővé teszi eltérő adattípusok csoportjainak egy egységként való kezelését. Az ilyen típusú programokban szükség lehet arra is, hogy a program bizonyos szakaszaiban más-más típusú adatokat tároljanak ugyanazon a memóriaterületen. Ezért a Modula-2 rekordjaiban lehetőség van ún. variáns részek deklarálására, amelyek egy szelektormező tartalmától függően a rekord egy-egy adott részének különböző kezelését teszik lehetővé.

A rekordtípus a legmegfelelőbb például a komplex számok, a dátum, az idő, a személyi adatok stb. kezelésére. Egy személyt leírhatunk a család- és keresztneve, valamint születési dátuma segítségével. Az ilyen struktúrák tipikusan hierarchikus elrendezésűek, és mivel a dátum is hierarchikus felépítésű, ezért a személyt jellemző struktúra kétszintű. Az ilyen hierarchikus adatstruktúrákat könnyen deklarálhajuk rekordtípusok segítségével. A rekordstruktúra lehetővé teszi, hogy akár a teljes adatstruktúrát, akár annak egyes elemeit „vegyük kézbe". A rekord elemeinek elnevezése mező, nevük pedig a mezőazonosító. Ezek az elnevezések a hagyományos, táblázatszerű adatbázis-reprezentációból származnak (lásd a „Háttérinformáció" alatt). Minden egyes mezőnek típusa van, és ezek a típusok általában nem megegyezőek. A rekordtípus mezőinek deklarációja hasonlít a változók deklarációjára. Míg a tömböknél a t tömb i. elemét t[i]-vel jelöljük (a tömb azonosítóját követi az index), addig rekordok esetén r rekord f mezőjére az r.f kvalifikált azonosítóval hivatkozhatunk (a rekord azonosítóját követi a mező azonosítója).

A Modula-2-ben lehetőség van tetszőlegesen strukturált típusok használatára a mezők deklarálásakor, így bonyolult struktúrahierarchiát hozhatunk létre. Ez azzal jár, hogy a hivatkozások maguk is szekvenciálisak lesznek: például p.first[3] a példák deklarációi alapján egy adott személy (p) családnevének 4. karakterére történő hivatkozás, míg p.birth.year ugyanennek a személynek a születési évét jelöli.

Miért és mikor nyerő a rekord?

Első pillantásra a rekord általánosított tömbnek tűnhet, mivel elemeinek nem kell azonos típusúaknak lenniük. Azonban a rekordoknál — a tömböktől eltérően — a hivatkozáshoz fix azonosító kell, míg tömbök esetében az index tetszőleges kifejezés lehet: jellemzően számítások eredménye (feltéve persze, hogy az eredmény típusa megegyezik az index típusával). Azonos típusú rekordmezők a program értelmezését sok esetben könnyebbé teszik, mint ha tömböket használtunk volna. Például:
Complex = RECORD re,im:REAL END — ez sokkal olvashatóbb, mint Complex = ARRAY [0..1] OF REAL (mivel z.re nyilvánvalóvá teszi, hogy z valós részéről van szó, míg z[1] nem).

Hasonlóképpen, a
Vector = RECORD x,y,z:REAL END deklaráció egyértelműbb, mint a Vector = ARRAY [0..2] OF REAL (és a v.x, v.y, v.z jelölésmód is egyértelmű).

A Person = RECORD first,last:ARRAY [0..15] OF CHAR definícióban

Névjegy

Rekord

A rekord mint összetett típus eltérő típusú változók (elemek vagy komponensek) gyűjteménye, amelyekre egyetlen azonosítóval hivatkozhatunk. A rekordot felépítő elemeket mezőknek nevezzük. Egy mezőben a rekord deklarálásakor meghatározott típusú értéket tárolhatunk. A Modula-2-ben a mező típusa tetszőlegesen komplex típus lehet (egész szám, karakter, tömb, rekord stb). A deklarációban minden egyes mezőhöz mezőazonosítót rendelünk, és ezekkel az azonosítókkal hivatkozhatunk a rekord egyes elemeire.

A rekordokat jellemzően olyan esetekben használják, amikor egy adott objektum vagy tárgy (például személy, árucikk) különböző tulajdonságait (például a személy neve, személyi száma és neme, illetve az árucikk cikkszáma, megnevezése és ára) összefogottan akarják kezelni. A rekordok legelterjedtebb alkalmazási területe éppen ezért az adatbázis-kezelés.

A Modula-2 rekordokban lehetőség van variáns rekordok és rekordmezők deklarálására. A variáns rekordok lehetővé teszik az optimalizált memóriakihasználást, a típuskonverziók követhető megkerülését és egy adott bináris reprezentáció értelmezését különböző módokon.

A rekordokat gyakran használják mutatókkal együtt dinamikus struktúrák létrehozására.

Példának okáért

Rekordtípusok deklarációja:

```
TYPE
  Date = RECORD
           year : [1900..2100] ;
           month : [1..12] ;
           day : [1..31] ;
         END ;
  (* variáns rekordtípus *)
  Person = RECORD
             first, last : ARRAY [0..30] OF CHAR ;
             age : SHORTCARD[0..150] ;
             CASE sex : Sex OF
             | Female : children : SHORTCARD ;
             | Male : militaryService : BOOLEAN ;
                  militaryGrade : Grade ;
             END (* CASE *) ;
             address : ARRAY [0..40] OF CHAR ;
             birth : Date ;
           END ;
  (* egymásba ágyazott rekordok *)
Hivatkozás:
p.first := "Smith" ;
ReadCard ( d.year ) ; (* paraméter a rekord egy eleme *)
ReadPerson ( p ) ; (* paraméter az egész rekord *)
További példák az m2-11.lst fájlban.
```

a családnév és a keresztnév jól elkülönülnek, míg Person = ARRAY [1..2] OF ARRAY [0..15] OF CHAR esetben ez már koránt sincs így. Ezenfelül a rekordtípus esetén lehetőség van a két nevet eltérő hosszúságú tömbökben tárolni, míg a kétdimenziós tömb esetén ez nem lehetséges.

A rekordok a mezőértékek tetszőleges kombinációját tartalmazhatják, így például a Date típus esetén a month = feb és day = 31 kombináció megengedett, bár mint dátum nem érvényes. Ez matematikailag azt jelenti, hogy egy rekordtípus szorzathalmaz, azaz például egy Date típusú változó 31*12*201=74772 értéket vehet fel, amelyek között szerepel például 1992 február 31. is.

A Modula-2 a rekordtípusokra a következő operátorokat értelmezi azonos rekordtípusú operandusok esetén: értékadás, egyenlőségvizsgálat, különbségvizsgálat. Minden más művelet definiálása a programozó feladata.

Variáns rekordok

Bizonyos esetekben egy rekord néhány mezője csak akkor érvényes, ha egy másik mező valamilyen meghatározott értéket vesz fel. Nézzünk két példát!

Vegyük, hogy egyszerű könyvtári adatbázis kezelése a feladat. Az adatbázis a címet, a szerzőt, a kiadvány típusát (könyv, szakdolgozat vagy újságcikk) minden esetben tartalmazza. A könyvek és szakdolgozatoknál a kiadó, míg az újságcikkeknél az újság neve, az évfolyam és az oldalszám szerepel az adatbázisban.

A másik esetben geometriai alakzatokat szeretnénk Modula-2 változókként használni. Az alakzatoknak a következő attribútumai vannak: szín (piros, zöld, kék vagy sárga); forma (négyzet, kör, háromszög vagy szakasz); valamint egyéb tulajdonságok, amelyek leírják az alakzatok elhelyezkedését és orientációját. Minden alakzatnak van színe és formája, de a geometriai tulajdonságokat már eltérően kell leírni.

Mindkét szituációban célszerűnek látszik rekordtípus deklarálása, azonban a fix felépítésű rekordok nem felelnek meg 100%-osan az igényeinknek, és az esetlegesen kialakított rekordstruktúra helykihasználása vagy inkább helyfoglalása sem lenne optimális. A Modula-2 az ilyen jellegű feladatok megoldására vezeti be a variáns rekordtípusokat. Ezekben a rekordtípusokban a variáns részek egy fix rekordmező értékétől függenek. Ennek a fix mezőnek a neve szelektormező. A szelektormező értékeitől függően a variáns rész különbözőképpen definiálható. A szelektálást meghatározó, úgynevezett értékleírások a szelektormező típusával megegyező típusú, át nem lapolódó állandó kifejezések.

Szelektálás a variánsokból

A variáns rész deklarációjában lehetőség van a külön nem szereplő értékle-

Pontosan és szépen

Definíció

A rekord fix számú komponensekből álló strukturált adattípus. A rekord elemei tetszés szerinti típusúak. A rekord deklarációja minden elemre — az úgynevezett mezőkre — definiál egy típust és egy mezőazonosítót. A mezőazonosítók láthatósága a rekordon belülre korlátozódik. Ezért az egy rekordon belüli mezőazonosítóknak eltérőeknek kell lenniük, viszont nem kell különbözniük a rekordon kívüli azonosítóktól. A mezőkre hivatkozni a mezőazonosítókkal vagy kvalifikált azonosítóval, vagy — WITH utasításokban — magával a mezőazonosítóval lehet. Egy rekordváltozó minden mezőjében egy, a mező típusának megfelelő értéket tartalmaz.

Egy rekordtípus tartalmazhat variáns részeket is, mezőcsoportok alternatív használatára. Ebben az esetben az adott rész első mezője az úgynevezett szelektormező. A szelektormező értéke jelzi, hogy az adott rész mely változatnak felel meg. A szelektormező egy sorszámozott, szelektortípusú mezőnek felel meg. Az egyes variáns részeket értékleírásokkal azonosítjuk. Ezek az értékleírások a szelektormező típusának megfelelő konstans kifejezések. Az értékleírások lehetnek intervallumok is, de nem tartalmazhatnak átlapolódó értékeket. A szelektormező elhagyása esetén az értékleírásoknak csak szintaktikai szerepük van. Egy rekord tetszőleges számú variáns részt tartalmazhat, illetve a variáns részek tetszőleges mélységben egymásba ágyazhatók. A TopSpeed Modulában lehetőség van eltérő variáns részekben megegyező azonosítók használatára, ha a mezők rekordon belüli offsetje megegyezik.

Formálisan:
```
$ RekordTípus = RECORD MezőLista-Szekvencia END.
$ MezőListaSzekvencia = MezőLista {";" MezőLista}.
$ MezőLista = [IdLista ":" Típus |
        CASE [Id] ":" kvalident OF
        variáns {"|" variáns}
        [ELSE MezőListaSzekvencia] END].
$ variáns = [ÉrtékleírásLista ":" MezőListaSzekvencia].
$ ÉrtékleírásLista = Értékleírás {"," Értékleírás}.
$ Értékleírás = ÁllandóKifejezés [".." ÁllandóKifejezés].
```

Pascal és C

A Pascal és a Modula-2 rekordtípusai között néhány kisebb eltérés van. A Pascalban a variáns rekordokban csak egyetlen variáns rész szerepelhet, és ezt meg kell előzze minden fix mező. A variáns részek értékleírásaiban a Pascalban állandó, míg a Modula-2-ben állandó kifejezés állhat. A Modula-2 rekordtípusai és a C struktúratípusai között nincs eltérés. A Modula-2 variáns rekordjai és a C uniótípusai között a következő összefüggések vannak: A C egy uniótípusa egyetlen variáns részből álló, fix mezőket nem tartalmazó, szelektormező nélküli Modula-2-beli variáns rekordnak felel meg. A C-ben lehetőség van struktúrák és uniók kombinálásával Modula-2-szerű variáns rekordok létrehozására (lásd az m2-11.lst fájlt). A C struktúráira és unióira semmilyen operátor sincs értelmezve, még az értékadás sem. A K&R C még a struktúrák mint paraméterek átadását sem engedi meg. Ezeket a hiányosságokat az ANSI C-ben már igyekeztek kiküszöbölni.

írások együttes kezelésére is. Eseteinkben a fix mezők írják le a közös tulajdonságokat, amelyek közül az egyik (a szelektormező) értéke alapján a speciális tulajdonságok a variáns részben találhatók.

Szótár

Rekord: Összetett adattípus, amely lehetővé teszi, hogy eltérő típusú, de logikailag összetartozó értékeket egyetlen változóban csoportosíthassunk.
Mező: A rekord elemeinek (részeinek) elnevezése.
Variáns rekord: A rekordok olyan típusa, amely lehetővé teszi, hogy egyazon memóriaterületen — különböző időpontokban — eltérő típusú értékeket tárolhassunk.

Mivel a legtöbb Modula-2 implementáció a variáns mezőkre hivatkozáskor nem ellenőrzi a szelektormező értékét, hogy elkerüljük a hibákat, a hivatkozás előtt magunknak kell megvizsgálni a szelektormező tartalmát. Bizonyos esetekben (memóriaspórolás, típuskonvenciók megkerülése, egyéb furfangok) szükség lehet a szelektormező elhagyására, ilyenkor csak egy szelektortípust definiálunk a variáns részhez, ami gyakorlatilag csak szintaktikai szerepet tölt be. Az adott típus megválasztása legtöbbször csak az előfordulható variánsok számától függ. Természetesen ilyen esetben nincs arra lehetőség, hogy a szelektormező tesztelésével meggyőződjünk arról, hogy éppen melyik variáns az érvényes. Itt jegyzem meg, hogy a mágneslemez mellékleten található m2-10.lst fájl előző havi lemaradásomat pótolja, mivel a tömbökkel kapcsolatos tudnivalókat példákkal illusztráló listafájl az én hibámból nem került el a szerkesztőségbe. A rekordokkal az m2-11.lst listafájl és a példaprogramok foglalkoznak. A programok a rekordok használatát mutatják be képernyő, billentyűzet, DOS/BIOS rendszerhívások és adatbázisfájlok (dBASE) kezelésekor. Az e havi utility modul az előző lemezmellékleten szereplő játékprogram továbbfejlesztett változata. A játékosnak ebben az új verzióban már ellenfelei is akadnak — falánk rekordok személyében. Aki ennél többet akar megtudni, böngéssze a forráskódot, annál is inkább, mivel rossz szokásomhoz híven ez a program sem ismerteti a működését (talán majd a végleges variáns!!).

Villányi László

Tények, tippek, trükkök

Egy rekordtípus mezőazonosítóinak mindig eltérőeknek kell lenniük, még akkor is, ha különböző variáns részben szerepelnek.
A TopSpeed Modula-2 megengedi egy rekordon belül azonos azonosítók használatát, ha azok eltérő variáns részben vannak, és offsetjük a rekordon belül megegyezik.
A variáns rekordok méretét a lehető legnagyobb méretű variáns elrendezés adja.
A szelektormező használata segít az éppen érvényes variáns kiválasztásában — így elkerülhető, hogy a rekordon olyan variáns részre történjék hivatkozás, amely az adott pillanatban nem érvényes. (Sok Modula-2 implementáció ugyanis nem ellenőrzi a szelektormező értékét a variáns mezőkre hivatkozáskor — lásd az m2-11.lst fájlt.)
Ha nem deklarálunk szelektormezőt, akkor nincs lehetőségünk feltételvizsgálatra.
Bár megengedett, de programozástechnikai szempontból nem célszerű olyan mezőazonosítót választani, amelyet a programban már másképpen deklaráltunk. Ez alól kivétel lehet az az eset, amikor eltérő rekordtípusokban azonos funkciójú és típusú mezőket deklarálunk, ezzel is kiemelve a hasonlóságot.
Azonos rekordtípusú változókra a Modula-2 automatikusan értelmezi az egyenlőség, a különbség és a hozzárendelés operátorokat.

Háttérinformáció

A rekordtípusokat az alkalmazások nagy többségében valamilyen adatbázis-szerkezet reprezentálására használják. Ezek lehetnek rekordokból álló tömbök, de nagyobb adatmennyiség esetén ez lehet fájlban tárolt adatok halmaza. Maga a rekord elnevezés is az adattárolás terminológiájából származik.
Az adatbázis-kezelés a számítógépes alkalmazások szinte mindegyikében jelen van. A programok nagy része adathalmazokon végez műveleteket, hogy újabb adatokat generáljon, majd ezeket a be- és kimeneti adatokat kezeli. Ezek közül a programok közül az adatbázis-kezelők a legelterjedtebbek. Az adatbázis-kezelő rendszerek nagy adathalmazokra vonatkozó műveletekkel foglalkozó programok. Ezek a rendszerek kizárólag az adatok karbantartására valók, mégis a legfontosabb számítógépes alkalmazások, mivel a számítógép a hagyományos kézi módszernél nagyságrendekkel gyorsabban képes óriási adathalmazokat kezelni. Ott, ahol a számítógépes kultúrának nagyobb hagyományai vannak, mint nálunk, szinte minden adat elérhető valamilyen számítógépes adatbázisból. Mivel a tudás (az információ) hatalom, és az idő pénz, ezért érthető, hogy az adatbázis-kezelő rendszereket nagy becsben tartják. Maga az adatbázis nem más, mint egy (sok esetben meglehetősen nagy) táblázat, oszlopokkal és sorokkal. A táblázat sorait rekordnak hívják. A rekordok mezőkből állnak. A mezők a rekord által reprezentált információ egy-egy részét tartalmazzák. Minden mezőnek van neve. Az azonos nevű mezők alkotják a táblázat oszlopait. Minden adatfájlnak van egy rekordformátuma, mely leírja a mezők neveit és sorrendjüket a rekordokon belül. A Modula-2 rekordtípusa egy az egyben megfelel az adatbázis-technikában használt rekordfogalomnak és -terminológiának.

Kalózperspektívák északi szomszédunkban

Munkában a szoftverrendőrség

Cseh-Szlovákiában a szerzői jogi törvényt csak 1990 márciusában módosították úgy, hogy annak hatásköre a szoftverkészítőkre is kiterjedjen, majd megalakult az Aura-Pont, amelynek szoftverjogvédelmi osztályáról szóló hírek — konkrét intézkedései és a Computerworld c. lap számában Kalózcsapdák címmel megjelent írás alapján — külföldre is eljutottak. A többnyire csak „szoftverrendőrség" névvel emlegetett Aura-Pont igazgatóhelyettesétől, Zbynek Loebltől kaptunk bővebb tájékoztatást erről a szervezetről.

Az Aura-Pont ügynökség elsősorban irodalmi művek szerzői jogi kérdéseivel foglalkozik — színházak, kiadók és egyéb kulturális intézmények megbízásából. Zbynek Loebl kezdeményezésére egészítették ki profiljukat a szoftverek jogi védelmével. Először elkészítették a „Jelentés a számítógépes kalózkodás helyzetéről Cseh-Szlovákiában" c. összefoglaló tanulmányt, majd létrehozták a jelenleg is irányítása alatt álló osztályt, azzal a céllal, hogy a számítógépes programok, adatbázisok és más szoftvertermékek szerzőinek, illetve a szerzői jogok tulajdonosainak szolgáltatásokat nyújtsanak. Az államügyészség engedélyével terveznek néhány konkrét ellenőrző akciót, melyekből eddig kettőt bonyolítottak le.

Az Aura-Pont segít ügyfeleinek összegyűjteni a szerzői jogaikat sértő szoftverkalózkodás bizonyítékait, elkészíti a szakértői véleményeket, és képviseli őket a peres eljárásokban. A megfelelő előírások és a megbízási szerződésben rögzített elvek alapján ellenőrzéseket hajt végre kiállításokon, szoftverbörzéken és más nyilvános rendezvényeken. A kalózkodásról kapott (nem névtelen) bejelentéseket továbbítja a szerzőknek vagy a szerzői jogok tulajdonosainak, s lehetősége van a további vizsgálódások koordinálására is.

A szerzői jogok érvényesítését elősegítő más szolgáltatásokat is szerveztek, például programok regisztráltatását és ún. „szerződéses ellenőrző rendszereket" — olyan cégek közreműködésével, mint az Autodesk vagy Fox Software. Az ügyfelek az Aura-Pontnál regisztráltatott vagy elhelyezett hazai szoftvertermékek megjelölésére felhasználhatják a „Czechoslovak software" elnevezésű szerzőjogi védjegyet. (Az erre jogosító szerződés évi díja magánszemélyeknek 600 korona, jogi személyeknek 2500 korona.)

Az Aura-Pont kétféle fő céllal őrzi meg a számítógépes programok forráskódját és dokumentációját:

a) A forrásprogramot mikrofilmre fotózva kell leadni, amivel vita esetén bizonyítani lehet, hogy a szoftvert ki, mikor és hogyan készítette el.

b) A forrásprogramok mágneses adathordozón való őrzése arra szolgál, hogy azt a szerződésben meghatározott feltételekkel az arra jogosultak bármikor megkaphassák — például a szerző távollétében programhiba kijavítása céljából. (Az őrzési díj magánszemélyeknek programonként 450, jogi személyeknek 900 korona évente.)

A megőrzésnek nagy jelentősége lehet az illegális másolatok (és átírt verziók) felfedésekor, vagy a programozó és a munkaadó közötti viták eldöntésében is, amikor például azt kell kideríteni, hogy a program mely részei készültek munkaidő alatt, s melyek azon kívül. (Ugyanis a szerzőnek csak az utóbbi esetben van a használatához és a terjesztéséhez korlátlan joga.) A megőrzés időtartama 3 év, s annak leteltekor további 3 évvel meghosszabbítható.

Az ügynökség képviseli a szerzőket honoráriumaik inkasszálásánál. Ez kiterjed az irodalmi és művészeti munkát érintő adólevonásra, a kulturális alaphoz való 2%-os hozzájárulásra és a nettó honorárium átutalására. A közvetítésből adódó rezsiköltség a bruttó honorárium 3%-át teszi ki. E szolgáltatásnak része a szoftverszerződések összeállításához nyújtott jogi segítség is.

Az Aura-Pont együttműködik a Cseh-Szlovák Kriminalisztikai Társasággal és más szervezetekkel. Egy kassai cégtől közös vállalat alakítására is kaptak ajánlatot. Arra törekszenek, hogy fizetett vagy önkéntes ellenőrökből álló országos hálózatot hozzanak létre, a leginkább érintett cseh-szlovák programkészítőkkel szorosan együttműködve.

A kalózkodás ellen alkalmazott módszerek közé tartoznak mindenekelőtt a felvilágosító kampányok, a jogi normák tökéletesítése és a konkrét ellenőrző akciók. Azzal is tisztában vannak ugyanakkor, hogy a harc kulcsa a programkészítők kezében van: a felhasználók igényeit kielégítő programokkal és a megfelelő szintű kiegészítő szolgáltatásokkal sokkal több eredmény érhető el, mint az ellenőrzésekkel, bírósági eljárásokkal. Mindazonáltal ahhoz, hogy a szerzők munkájával, és a cégek beruházásaival ne lehessen visszaélni, egyaránt szükségesek a jogi szankciók és a meggyőzést elősegítő módszerek.

A számítógépes szoftvertermékek Cseh-Szlovákiában a szerzői jogokat rögzítő 247/90 számú törvénycikkely védelme alatt állnak. A szerző vagy a szerzői jogok tulajdonosa igényelheti a jogtalan magatartás megtiltását, a jogsértés következményeinek eltávolítását, megfelelő kártérítést, és az illegálisan szerzett anyagi haszon kiadását. A szerzői jogok tudatos megsértése a büntető törvénykönyv 152-es paragrafusa alapján egymillió koronás felső határú pénzbírsággal, vagyonelkobzással illetve egy évig terjedő szabadságvesztéssel sújtható.

A szerző jogi törvényen kívül a szoftverek mint „know-how" termékek a gazdasági törvénykönyv alapján is védettek. Az előkészítés alatt álló kereskedelmi törvénykönyv ezt a módosítást az üzleti titok védelméről szóló paragrafus módosításával kívánja helyettesíteni, amely majd a számítógépes programokra is vonatkozik.

Puntigán József

Ha az alkotás piacképes

Szerzői jog és versenyjog

A szerzői jogi szabályozás erőteljesen befolyásolja a szerzői alkotásokat felhasználó vállalkozók versenyét. A szoftver bonyolult jogi helyzetének megértéséhez érdemes történelmileg messzebbre visszapillantva is szemügyre venni, miként illeszkednek egymáshoz a szerzői jog és a versenyjog fő rendelkezései.

A szerzői jog a piaci versenynek köszönheti létét és kibontakozását. Gutenberg 1450 körül született kettős találmánya (a szedhető betűk tartós ötvözete és a sajtógép) eleinte nem is a születése helyén — Mainz városában és a környező, a XV. századi európai fejlődésben elmaradt mintegy 300 német feudális kisállamban — forradalmasította a könyvpiacot, hanem a tengeri kereskedelem előnyeit élvező olasz városállamokban. Velencében az egymás könyveit utánnyomó kiadók már 1469-ben öt évre szóló könyvnyomtatási kizárólagosságot, „privilégiumot" biztosították az oda költözött Johann von Speyernek. Később meghatározott művekre szóló, azok mások által történő utánnyomása ellen meghatározott ideig védő privilégiumokat kaptak a nyomdász kiadók.

Mikor azután — fejlődésük fokának megfelelően — előbb Angliában, majd Franciaországban és végül német területen a polgári forradalmak vetettek véget a feudális privilégiumoknak, a szerzői alkotásokat sokszorosító nyomdászoknak közvetlenül juttatott közjogi kizárólagosság helyébe a kiadott mű szerzőjétől szerződéssel származtatott kizárólagos jogok léptek.

Piac és rizikó

A kibontakozó korlátlan piaci verseny összeomlással fenyegette a kiadói ipart: az első megjelentetés rizikóját vállaló nyomdász befektetései alig térülhettek meg, míg a sikeresnek bizonyult művek célzott utánnyomásával telítődött a piac. Új mű kiadásának kockázatát egyre kevésbé lehetett vállalni. Kézenfekvő megoldás volt, hogy a kiadók egymás közti versenyét a XV. századtól kezdve kiadói monopóliumokkal, privilégiumokkal korlátozzák.

Az aránylag kevés privilegizált kiadó azonban nem tudta a növekvő keresletet kielégíteni, nem volt rá igazán szüksége, ezért rengeteg mű maradt kiadatlanul, és ismét terjedni kezdtek az illegális kiadások. Ezen segített a szerzői jog születése, azáltal, hogy valamely műre nézve annak szerzője lett a felhasználói kizárólagosság forrása. A felhasználói jogok megszerzésének rendjét ezek után az állam helyett világszerte a kulturális piac mechanizmusa szabta meg.

A kiadók versenye most már a valamely műre vonatkozó kizárólagos kiadói jognak a szerzőtől való elnyeréséért folyt. Mihelyt egy kiadó meghatározott időre kizárólagos felhasználási jogot szerzett, erre a műre nézve lezárult a versengés lehetősége. Ez a körülmény mindmáig arra ösztökéli a többi kiadót, hogy a még „gazdátlan" műve-

Szoftverjogi konferencia Prágában

1991. december 2–3.

Kereskedelmi gyakorlat az Egyesült Államokban — milyen a hatása Európában?
Bradford L. Smith, ügyvéd, Covington & Burling, London, az amerikai Business Software Alliance képviselője.

A szoftverterjesztés alapelvei Nagy-Britanniában.
Christopher Reed, az információs jog tanára, Queen Mary & Westfield College, Londoni Egyetem; a „Computer Law & Practice" c. lap kiadója, több szoftvercég jogi tanácsadója.

A német szoftverpiac, avagy a szoftverkereskedelem ott, ahol a Kelet és Nyugat találkozik.
Thomas J. Hoeren, a Münsteri Egyetem tanára.

A magyar szoftverpiac jogi és üzleti specifikumai.
Pálos György, a budapesti InterCodex & InterLicence igazgatója; az AIPPI tanács tagja, a CISAC törvényhozó testületének vezetőségi tagja.
Oberrecht Gabriella, InterCodex & InterLicence, Budapest.

Szoftverterjesztés a skandináv országokban.
Andreas Galtung, a Norvég Információjogi Kutatóintézet szakértője.

A szoftverek eredményes terjesztésének lehetőségei Kelet-Európában.
Martin Bohácek, a Prágai Műszaki Főiskola jogi tanszékének tanára.
Zbynek Loebl, a prágai Aura-Pont igazgatóhelyettese.

Az EGK bizottságának szerepe a gazdasági verseny védelmében és az ezzel összefüggő szoftverszerződések irányításában.
Thomas C. Vinje, ügyvéd és igazgató, Morrison & Foerster, Brüsszel.

A szoftvergyártók és felhasználók álmai és igényei.
H. W. F. Borgerhoff Mulder, ügyvéd, nemzetközi jogi tanácsadó, Lotus Development European Corporation.

Szoftverszerződések a szovjet cégekkel — a jogvédelem fő formái.
Irina V. Savelyeva, a szellemi javak jogi védelmének tanára, Moszkvai Állami Egyetem; az S. J. Lex International, moszkvai magáncég igazgatóhelyettese.

Szoftverterjesztés Olaszországban.
Vittorio Frossini, a szellemi értékek jogi védelmének tanára, „La Sapienza" Egyetem, Róma.

További felvilágosítást a prágai szervezők adnak:
Telefon:+42 2 231 5015 Fax: +42 2 232 5334

kért versengjenek, s ezzel új, piacképes alkotásokra ösztönözzék a szerzőket. A nem kizárólagos felhasználói jogok esetében pedig a verseny az azonos műveknek a fogyasztókhoz való eljuttatásában is kibontakozhat. Mindez igaz a számítógépi programok alkotására és piaci felhasználására is.

A szerzői jog a fejlett piacgazdálkodásban a szerzőknek az alkotótevékenységükkel megalapozott személyiségi és alkotói jogait védi. A nekik biztosított kizárólagos jogok alapján válhat a mű a felhasználás menetében piaci áruvá, függetlenül attól, hogy esetenként kizárólagos vagy nem kizárólagos felhasználói jog alapján értékesítik azt. A felhasználók versenytársi érdekeit azonban már nem csupán a szerzői jog (Magyarországon az 1969. évi III. törvény — a továbbiakban Szjt — és a végrehajtásáról rendelkező 1969. évi MM sz. rendelet) védi, hanem a tisztességtelen piaci magatartás tilalma is (1990. évi LXXXVI. törvény — a továbbiakban Tvt).

A történeti fejlődés azt mutatja, hogy a szellemi alkotások piaci felhasználása csak akkor lehet eredményes, ha azt meghatározott időre kizárólagos szerzői jog biztosítja. A kizárólagos szerzői jog és annak a szerzők képviseletében, központi szervezet útján történő gyakorlása a mű tömeges felhasználásakor (hiszen például zeneművek nyilvános előadásain az esetenkénti jogosítás gyakorlatilag lehetetlen) összhangban áll a versenyjog egyik alapelvével. E szerint a gazdaságilag indokolt közös célok eléréséhez szükséges mértéket meg nem haladó versenykorlátozás vagy kizárás nem esik tilalom alá, ha az azzal járó előnyök meghaladják az azzal járó hátrányokat (Tvt 17. szakasz).

Azonosítás és integritás

A szerzői jognak a mű felhasználásával kapcsolatos elvárásai sok esetben egybecsengenek a felhasználók által terjesztett műpéldányokkal kapcsolatos versenyszabályokkal. Így például az Szjt 9. §-a szerint a szerzőt megilleti a jog, hogy művén szerzőként feltüntessék; a szerző követelheti, hogy e minőségét senki ne vonja kétségbe. Ez a szabály a szerzői alkotás vonatkozásában egészíti ki a Polgári Törvénykönyv 77. szakaszának rendelkezését, amely szerint mindenkinek joga van a név viseléséhez. A Polgári Törvénykönyv ehhez hozzáfűzi azt is, hogy a névviselési jog sérelmét jelenti különösen, ha valaki jogtalanul más nevét használja, vagy máséhoz hasonlót használ. A szerzőnek elemi érdeke fűződik ahhoz, hogy saját művével kapcsolatban feltüntessék szerzői mivoltát, ugyanakkor idegen mű jelölése során ne keltsék azt a látszatot, mintha azt ő alkotta volna.

A mű összetéveszthetőségével szembeni szerzői jogi védelemnek felel meg a mű felhasználója által forgalmazott műpéldányok összetéveszthető jelölésének tilalma is. A Tvt 7. szakasza szerint: (többek között) tilos az árut olyan jellegzetes megjelöléssel vagy elnevezéssel forgalomba hozni, továbbá olyan nevet, megjelölést vagy árujelzőt alkalmazni, amelyről a versenytársat, illetőleg annak jellegzetes áruját szokták felismerni. Az első, 1923-ban kibocsátott, a tisztességtelen versenyt tiltó magyar törvény kifejezetten is tiltotta valamely könyv címének a versenytárs kiadó általi átvételét. A címvédelemről mai jogunkban csak a szerzői jogi törvény 13. szakasza rendelkezik, amely szerint a szerző hozzájárulása szükséges a mű sajátos címének felhasználásához is.

Az Szjt és a Tvt említett rendelkezései a mű, illetőleg a többszörözött műpéldányok azonosítása tekintetében a szerző és a mű felhasználója érdekében kölcsönösen kiegészítik egymást. Ezek a szabályok megfelelően alkalmazandók a számítógépi programok, illetőleg a forgalmazott programhordozók tekintetében is.

Az Szjt 10. szakasza szerint a szerző jogát sérti művének bármilyen jogosulatlan megváltoztatása. Nem jogosulatlan az, amihez a szerző hozzájárult. Ez a rendelkezés a szerző művének integritásához fűződő érdekeit védi. A fogyasztónak is van azonban jogvédte érdeke azzal kapcsolatban, hogy a művet hordozó, forgalmazott műpéldányok valóban az eredeti művet közvetítik-e hozzá, vagy annak csupán rövidített, illetőleg megváltoztatott mutációját. A Tvt a fogyasztók megtévesztésének tilalma körében idevágó rendelkezéseket is tartalmaz. A 11. szakasz /2/ bekezdés a) és c) pontjai értelmében tilos megtévesztésnek minősül különösen, ha az áru lényeges tulajdonsága, továbbá eredete tekintetében valótlan tényt állítanak, vagy az áru lényeges tulajdonságairól más, megtévesztésre alkalmas vagy hiányos tájékoztatást adnak. Továbbá, ha elhallgatják azt, hogy az áru nem felel meg az azzal szemben támasztott szokásos követelményeknek.

Az áru eredetére nézve valótlan tényállítás vagy hiányos, illetve megtévesztésre alkalmas tájékoztatás az is, ha az abban foglalt mű szerzőjének nevét mellőzik, vagy pontatlanul jelölik meg, esetleg más nevének említése által. Az áruval szemben támasztott szokásos követelményektől való eltérés elhallgatásának minősül, ha valamely könyvön, programhordozón, más műpéldányon nem tüntetik fel, hogy az csupán kivonatos, nem teljes vagy esetleg részben módosított művet tartalmaz.

Erőfölény és méltányosság

A szerzők jogvédelme szempontjából említést érdemel a Tvt 4. fejezete, amely a gazdasági erőfölénnyel való visszaélés tilalmát tartalmazza. Ebben a körben kiemelem a 20. szakasz néhány rendelkezését, amelyek szerint tilos a szerződéses kapcsolatokban egyoldalú előnyt kikötni, továbbá hátrányos feltételek elfogadását kikényszeríteni, vagy a szerződéskötéstől indokolatlanul elzárkózni.

Többnyire gazdasági erőfölényben van a szerzővel szemben az, aki a szerző művét a fogyasztókhoz közvetíti, eljuttatja.

Kezdő vagy kevésbé népszerű szerzők, a jogi és kereskedelmi lehetőségeiket illetően kevésbé tájékozott alkotók már csak azért is függő helyzetbe kerülhetnek a műveik terjesztésére vállalkozóktól, mert nélkülük aligha tudnák műveiket a közönség elé tárni. Fokozódik ez a gazdasági erőfölény akkor, ha meghatározott műfajú művek forgalmazására a kulturális piacon nagyon kevés számottevő cég akad, különösen ha azok együttes piaci részesedése az 50%-ot is meghaladja.

Indokolatlan, egyoldalú előnynek minősül a Tvt 22. szakasza szerint, ha a szerződésben kikötött szolgáltatás és ellenszolgáltatás között feltűnően nagy az értékkülönbség. Ennek megállapításához vizsgálni kell a szerződéskötés körülményeit, a szerződés egész tartalmát, a forgalmi viszonyokat, az értékviszonyokat, az ügylet jellegéből fakadó sajátosságokat, valamint a szolgáltatás és ellenszolgáltatás meghatározásának módját.

A Tvt mögöttes jogként az Szjt és végrehajtási rendeletének a művek felhasználására kötött szerződésekre vonatkozó passzusait egészíti ki. A gyakorlati szerzői jogvédelemnek talán legfontosabb követelménye, hogy a jog egyértelműen határozza meg a szerzők és a műveiket felhasználó vállalkozók érdekegyensúlyának garanciáit, amelyekre a felhasználói szerződések tartalmának kialakításakor tekintettel kell lenni.

Boytha György

DataEase
INTERNATIONAL

A **DataEase** adatbázis-kezelőt azoknak ajánljuk, akik értik az: angol, dán, finn, francia, holland, izlandi, magyar, német, norvég, olasz, orosz, portugál, spanyol, svéd nyelvek valamelyikét, ugyanis a **DataEase International** terméke ezeken a nyelveken is tud.

A **DataEase** egy egyedi vagy többfelhasználós (LAN) adatbázis-alkalmazást fejlesztő rendszer DOS környezetben, azoknak, akik a saját szakmájuk szakértői, akik színvonalas alkalmazásokat kívánnak egy-két nap alatt létrehozni, akik egyszerű nyilvántartásokat készítenek munkájuk segítéséhez, vagy akár azoknak, akik a számítástechnika professzionális alkalmazói.

Angliában 1990-ben a PC-s relációs adatbázis-kezelők közül a vásárlók több mint 30 százaléka a **DataEase-t** választotta, jóval többen, mint akármelyik másik terméket.

A **DataEase International, Inc.** termékeinek magyarországi disztribútora a:

VT-SOFT Videoton Software Kft.
1033 Budapest, Vörösvári út 103-105.
Telefon: 180-3744
Telefax: 180-3750

VT-SOFT®
VIDEOTON SOFTWARE KFT

INFORMÁCIÓKÉRÉS: 36 ▼

KIVÁLÓ MINŐSÉG
24 HÓNAP GARANCIÁVAL

FAN computer

KÉSZPÉNZES ÁRLISTÁNKBÓL:
(Alapkonfiguráció: alaplap, 1,2 MB FDD, 44 MB HDD, 1s/1p/1g port, 102 g. billentyűzet, 14"-os mono. monitor, baby-ház)

AT-286/12/16 MHz	1 MB RAM		59 990,–
AT-386SX/16 MHz	2 MB RAM		78 570,–
AT-386/33 MHz	2 MB RAM	64 kB cache	115 430,–
AT-486/33 MHz	4 MB RAM	128 kB cache	189 930,–

Igény szerinti tetszőleges konfigurációk.

SecureData

MEMÓRIAKÁRTYÁS SZUPERBIZTONSÁGOS ADATVÉDELMI RENDSZEREK

FELLOW
KÖNYVMÉRETŰ ASZTALI SZÁMÍTÓGÉPEK
RÉSZEGYSÉGEK, „QUANTUM" WINCHESTEREK, MOUSE-OK, SCANNEREK, DIGITALIZÁLÓ TÁBLÁK

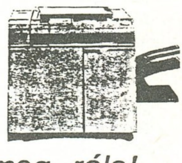
FAN

Tajvani—Magyar Vegyesvállalat
1118 Budapest, Késmárki u. 6.
(volt Friss István u.) Telefon/Fax: 185-0813

INFORMÁCIÓKÉRÉS: 16 ▼

Canon

VÁLTSON SZÍNESRE
Canon
MÁSOLÓGÉPPEL!
CSÚCS AMIT TUD:



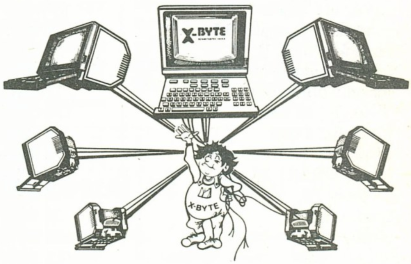
- Kicsinyítés
- Nagyítás
- Montírozás
- Tükörkép-készítés
- Képismétlés
- Poszter készítés
- 17 millió színáranyalat

Győződjön meg róla!

fénymásolás

TONER KFT
1095 Budapest, Mester utca 21.
Tel.: 113–1687, 134–3516

INFORMÁCIÓKÉRÉS: 03 ▼



a jövő most kezdődik!

X-BYTE®
SZÁMÍTÁSTECHNIKA
Törv. védve

SZÁMÍTÓGÉP HÁLÓZATOK
Kiváló minőség, közepes ár!

kiváló áruk fóruma

1138 Budapest, Népfürdő u. 17/E
Tel. és fax: 173-1232
Telex: 22-3399

Topreklám

INFORMÁCIÓKÉRÉS: 13 ▼

szoftver ABC

: 201-6891
201-2011/131
: 201-8619
: 1277 Budapest
23. Pf: 45.

A legjobbat a legjobbaknak, azonnal!

Raktárról szállított szoftvereink:

Szoftver:	Ár (+ÁFA)
Clipper 5.01	68 990,–
Corel Draw 2.0	50 900,–
Foxpro 2.0	61 490,–
Foxpro Lan	91 000,–
Foxpro Runtime	54 400,–
Framework IV	64 000,–
Harvard Graphics	48 000,–
Laplink V. 3.	12 900,–
MS-C 6.0	40 900,–
MS-DOS 5.0	8 400,–
MS-Excel 3.0	40 900,–
MS-MASM 6.0	15 200,–
MS-WORD for Xenix and Unix	99 900,–
MS-Windows 3.0	10 900,–
MS-Windows Device Dev. Sys.	35 900,–
MS-Windows (Szoftver) Dev. Sys.	35 900,–
MS-Word 5.5	33 110,–
MS-Word for Windows	41 490,–
Micrografx Charisma	37 900,–
Micrografx Designer 3.1	44 900,–
Nantucket Tools II	54 900,–
Norton Commander 3.0	12 000,–
Norton Utilities 6.0	13 000,–
Novell 2.2 10 user	130 000,–
Novell 2.2 5 user	58 900,–
Novell 386 3.11 100 user	475 000,–
Novell 386 3.11 20 user	230 000,–
PC Tools 7.0	12 000,–
Paradox 3.5	38 900,–
Printer Assist	16 900,–
Procom Plus	13 500,–
Quattro Pro 3.3	16 000,–
SCO Foxbase for Xenix	103 900,–
SCO Foxbase for Xenix Runtime	34 900,–
SCO TCP/IP for Unix 386	24 900,–
SCO Unix 3.2 Dev. Pack	78 900,–
SCO Unix 3.2 Op. Sys.	55 600,–
SCO Xenix 386 Op. System	70 600,–
SCO Xenix 386 Text Pr. Pack	24 900,–
Superbase 4 For Windows	50 900,–
Turbo Pascal for Windows	22 900,–
Ventura Publiser Gold for Windows	74 900,–
WordSTAR 6.0	36 900,–
Wordperfect 5.1	43 900,–

Még sok száz SZOFTVER rövid szállítási határidővel!

Kérje részletes katalógusunkat!

Érdeklődjön rendszeresen, a Szoftver ABC-ben mindig van akció!

INFORMÁCIÓKÉRÉS: 14 ▲

Ingres

VILÁGSZÍNVONAL HAZAI ÁRAKON

Ideális fejlesztőkörnyezet és futtatórendszer adatbázis-alkalmazásokhoz:

- **Komplett SQL adatbázis-kezelés.**
- **Negyedik generációs fejlesztőkörnyezet.**
- **Interaktív maszkeditor.**
- **Menü- és ablakkezelés.**
- **Tökéletes adatbiztonság.**
- **Tranzakció-kezelés.**
- **Nagy hatékonyság.**
- **Server–kliens architektúra.**

Újdonság!

Grafikus alkalmazásgenerátor (Windows 4GL) az SCO Open Desktop rendszeréhez.

Felvilágosítás:
VT-SOFT KFT.
Telefon: 180-3744
Telefax: 180-3750

INFORMÁCIÓKÉRÉS: 37 ▲

SZÁMÍTÁSTECHNIKA KULCSRAKÉSZEN!

A LEGKISEBB NOTEBOOK-TÓL A LEGGYORSABB 486-OSIG

- XT, AT, 386, 386SX, 486, Laptop minden kiépítésben.
- EPSON, STAR, NEC nyomtatók teljes választéka.
- MODEMEK és egyéb tartozékok széles választéka.
- Magánszemélyeknek KÉSZPÉNZFIZETÉS ESETÉN KEDVEZMÉNY!
- ASHTON-TATE, BORLAND, MICROSOFT, NANTUCKET, LOTUS szoftverek.
- SHAREWARE programok (1200-féle) 360,– Ft + ÁFA áron.
- MODEMES táv-adatátviteli és BBS rendszerek szállítása.
- VÍRUSÖLŐ program (120-féle vírust öl!)
- NOVELL HÁLÓZATI SZOFTVEREK, hálózatkiépítés.

Ajánlatunk:
NOTEBOOK SZÁMÍTÓGÉPEK MÁR 69 900,– Ft-tól

AT számítógép: 1 MB RAM, 40 MB HDD,
1,2 MB FDD, Mono 14" (PHILIPS)
1 S, 1 P, 101 gombos bill. 63 100,– Ft + ÁFA
(Készpénzért 59 900,– Ft + ÁFA.)

Amikor ezt a hirdetést Ön olvassa, áraink már úgyis alacsonyabbak! Ezért kérjük, telefonáljon vagy írjon, és mi örömmel adunk felvilágosítást, küldünk részletes árjegyzéket!

QWERTY

High Tech. Kft.
1117 Budapest XI., Orlay u. 4.
Telefon: 166-3098, 185-2687, Fax: 185-2687
BBS: 118-7950 BUDAPEST BBS

INFORMÁCIÓKÉRÉS: 38 ▲

Dokumentáció, óh!

Érdeklődéssel olvastam az Alaplap 1991. szeptemberi számában a szerzői jogokkal kapcsolatos cikkeket. Van azonban valami, amiről az írások keveset szóltak: a programok dokumentációi, illetve ezek szerzői jogi kérdései.

Mi is a helyzet a programleírásokkal? A legálisan vásárolt szoftverhez valamiféle leírás is jár. „Közkincs" jellegű programok esetén ez egyszerűen a lemezen található, ízlés szerint ki is nyomtatható szöveges állomány. Egy programnyelvhez, adatbázis-kezelőhöz vagy hasonló programrendszerhez viszont több vaskos kötet — felhasználói kézikönyv, referencia-kézikönyv stb. — is tartozik. Az illegálisan szerzett programhoz legfeljebb az eredeti dokumentáció fénymásolata — vagy csak a fénymásolat fénymásolata — szerezhető meg (egyébként elvileg a dokumentációk másolása is törvénybe ütköző cselekedet). Az eredeti kézikönyvek, leírások azonban többnyire angol nyelvűek. Örvendetes, hogy újabban több program magyar nyelvű változata is létezik (Ventura Publisher, dBASE IV stb.), értelemszerűen a hozzá-való magyar nyelvű dokumentációval együtt. Bizonyos szoftverek esetén a magyarosítás „veszélye" — éppen a szoftver lényegéből kifolyólag — nem fenyeget: ugyan hogy nézne ki egy C vagy Pascal nyelv magyar kulcsszavakkal?

A dokumentáció egyfajta formája a help alkalmazása. Az elterjedt programokhoz kiváló, helyzetérzékeny helpek állnak rendelkezésre. Véleményem szerint e helpek akkor hasznosíthatók igazán, ha a felhasználónak van valamiféle globális áttekintése arról, hogy az adott programrendszernek mik a lehetőségei, illetve korlátai. Ezután az online help jól használható, ha például egy konkrét eljárás meghívásakor annak paramétereit akarjuk tudni.

Összefoglalva a fentieket: nem biztos, hogy egyáltalán van dokumentáció, vagy ha van is, nem feltétlenül tud a potenciális felhasználó annyira angolul, hogy érdemben használni tudja. Ekkor jöhet az ötlet: nézzünk körül, mi az, ami elérhető magyar nyelven. A választék bőséges: az LSI ATSZ „szivárványfedelű" könyvei, a Műszaki Könyvkiadó Lapozgató Sorozata, SZÁMALK- és Novotrade-kiadványok, és így tovább. Bizonyos esetekben a bőség zavara jellemző. Hány könyv jelent már meg a dBASE-ről vagy a Turbo Pascalról? Igaz, hogy ebben a többféle elterjedt verzió is szerepet játszik. Az ilyen kiadványok kapcsán óhatatlanul eszembe jut: ha több könyv egyazon program ismertetését, használatát tartalmazza, akkor a könyvek szükségképpen legalábbis hasonló tartalmúak. Természetesen ez függ attól is, milyen jellegű programról van szó. Egy felhasználói program menüi, almenüi meghatározott sorrendet, belső logikát követnek, míg egy programnyelvhez saját ötletek sorát tartalmazó példatár állítható össze, a szintaktikai szabályok ismertetésén túl. Felmerül a kérdés: hol húzzuk meg a határt? Mikor mondhatjuk, hogy egy könyv lényegében az eredeti dokumentáció fordítása, és mikor, hogy önálló alkotásnak tekinthető? Egy, a Murphy-törvények közé illő „bölcsesség" jut eszembe: ha egy könyvből másolsz, akkor plagizálsz; ha kettőből, akkor összevonsz; ha háromból, akkor tanulmányt írsz; négy könyvből pedig megírhatsz egy ötödiket. Mi lehetne ezek után a megoldás? Elsősorban természetesen az, hogy mindenki legálisan jusson hozzá a használni kívánt szoftverhez. Ennek feltételei — árak és fizetések viszonya stb. — meglehetősen messze vezetnének. Jobb esetben a program és vele a dokumentáció is eleve magyarított változat, tehát nincs különösebb probléma. Más esetben célszerű angolul (is) tudni, nemcsak a számítástechnika, hanem az Európához való felzárkózás szempontjából is. Egy lehetőség, hogy megjelenhetnének az eredeti programdokumentációk magyar fordításai is. Ez utóbbi esetben egyértelműen megjelölve, hogy a könyv fordítás (és minek a fordítása), „copyright X. Y. Hungarian Translation" jelzéssel ellátva. E megoldás olyan szempontból is előnyös, hogy egy program adottságait a kifejlesztői ismerik leginkább, tehát feltehetőleg a legalaposabb, legrészletesebb leírást is ők tudják elkészíteni.

És addig, amíg bekövetkezik a fenti ideális állapot? Tartok tőle, hogy addig még várnunk kell egy kicsit.

Szegedi Gábor

Milyen nyelven beszél az „Akuku" vírus?

Kis János említi az Alaplap augusztusi számában — a szovjet vírusinvázióval kapcsolatban —, hogy az Akukunak becézett vírus a következő üzenetet írja be (egyébként latin betűkkel) minden megfertőzött programba:

„A kuku, Nastepny komornik!"

Mivel a vírusok keletkezési helyének és terjedési útjának kérdése mindnyájunk számára húsbavágó probléma, megpróbáltam nyelvi módszerekkel utána nyomozni. Íme az eredmény:

Elvileg elképzelhető lenne, hogy ez a vírusszöveg valami orosz nyelvű hadova, akkor azonban az orosz nyelv jegyeit kellene hordoznia. Ahogy például a fentebb említett halandzsaversről messzire lerí, hogy „magyarul van írva". (Az orosz nyelvészek is ismerik különben a halandzsa fogalmát: szállóigeként emlegetik például egyik ismert nyelvészük „Glokaja kuzdra..." kezdetű halandzsamondatát.) Itt azonban másról lehet szó. Két megoldás kínálkozik:

1. Amennyiben elírás vagy nyomdahiba a szövegben a „nastepny" szó p betűje és „nastenny" a helyes alak, akkor cseh (vagy szlovák) nyelvű a mondat, és így fordítható magyarra: „Kukucs! Itt a falon lakó albérlő!!!" Ha valóban Csehszlovákia és Ausztria felől érkeznek hozzánk szép számban a vírusok, akkor eléggé kézenfekvőnek látszik ez a magyarázat. (Az „albérlő" valami poloskaszerű élősködőre való utalás lehet ebben az esetben.)

2. Ha nincs elírás, akkor viszont a lengyel nyelv alapján adódó — szerintem jobban elfogadható — magyarázat a helyes: „Kukucs, itt a végrehajtó!!!"

Ami persze csak adalék, és nem bizonyíték a vírus származási helyének meghatározásához.

Vargha Dénes

Kaleidoszkóp

Augusztusi számunkban részletes elemzés jelent meg Farkas Ernő tollából két magyar „spelling checker" összehasonlításáról. Ha jól veszem ki az írásból, mindkét program, a „NyelvÉsz" is és a „Helyes-e" program is nagyon szűkre állítja az objektívjét, és környezetétől függetlenül vizsgálja az egyes szavak írásának helyességét. Ennek a megközelítésnek nyilvánvaló hiányossága, hogy nem tud dönteni környezetfüggő helyesírási kérdésekről. Melyik a helyes: AKKORA vagy AKKORRA? Attól függ. A maga helyén mindkettő helyes lehet — ennek eldöntésére azonban meg kell vizsgálni legalábbis a szó mikrokörnyezetét.

Játsszunk el a gondolattal, és fordítsuk meg a dolgot. Mit lehet kihozni magából a környezetből, a szavak ismerete nélkül? Pontosabban: milyen szerkezeti sajátosságok szűrhetők le a könnyen fölismerhető nyelvtani jegyekből, kötőszavakból, igekötőkből és egyéb véges listácskákkal felsorolható kvázinyelvtani jegyekből?

Ajánlom elemzésre például a következő „kvázimagyar" verset:

A RÉVLETEK HERÁNDOSAI

Szeládon fágva száll a máfor
Homondok: remélve feltereng,
S mi ketten, bús herándozók
Csak fégtünk, fántunk át a révleten.
Hahajhó! Visszamánt a táka!
Hahajhó! Lepönt a vádatás!
Nincs lardaság, csak fájó kérbedöntés,
Nincs éldelem, csak rétő álmarás.
Mi nem kárálva, nem sipogva bágunk,
Mi nem hemeghetünk rohátnokok
mögén,
Csak halk s homáló harsadások estén
Egybeferengünk ketten: te meg én.

(Hozzám a vers gyerekkoromban, szájhagyomány útján került. Alighanem Karinthyék kávéházi körében született annak idején, amikor — Karinthy humoreszkjeiből tudjuk — a „Kiszera méra bávatag" prózai szövegei mellett versírással is megpróbálkoztak az újonnan fölfedezett halandzsanyelven.)

A feladat a következő: programot kell készíteni a fenti szöveg (és hasonlóan egyszerű szintaktikai szerkezetű egyéb szövegek) plauzibilis elemzésére, amely nyelvtani pozíciójukból és egymáshoz való viszonyukból megállapítja az egyes szavakról, hogy

a) milyen szófajhoz tartoznak,
b) mi a szavak ragozatlan alakja,
c) mi a szavak belső tagolása, ha van (pl. igekötő, szótő, toldalék)?

Nem cél bonyolult szintaktikai szerkezetek fölismerése, de az értékelésnél előny, ha szabadon bővíthetők a program komponensei (szavak listái, szintaktikai szabályok vagy amit még szükségesnek ítél a feladat megoldásához a kedves olvasó).

A jutalom ugyanaz, mint az előző feladatnál, a határidő azonban lényegesen bővebb: november 30.

Vargha Dénes

Sorsolás — november végén

Sokan érdeklődtek már a potenciális nyertesek táborából: mikor tartjuk meg az ajándéksorsolást a lapunk kérdőívét beküldők között. A nyilvános sorsolásra

1991. november 27-én, délután három órakor

kerül sor a Cédrus Karolina Áruházban (Budapest XI., Karolina út 17.). Azért csak ekkor, mert a hivatalos formaságok betartásához, a közjegyzői hitelesítéshez szükség van az eredeti, beküldött kérdőívekre, s ezeket csak az értékelés befejeztével kaphatjuk vissza a munkát végző Cédrus Média BT-től.

Az értékelés tanulságaira még visszatérünk a decemberi Alaplapban, a nyertesek listáját — akik egyébként postán kapnak értesítést — januári számunkban közöljük.

A Mikrobazár rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás kereskedelmi tevékenységet folytatóknak gépelt soronként (60 karakter) 100 Ft, másoknak az első sor 50 Ft, minden további sor 20 Ft.

Kérjük, hogy a hirdetés díját a Cédrus Rt.-nek a Budapest Banknál vezetett 380-66760 számú számlájára utalják át, vagy postautalványon a Cédrus Rt. címére (1251 Budapest XI., Karolina út 17.) fizessék be, a hátoldalán feltüntetve, hogy apróhirdetés. A befizetést igazoló szelvényt a közlendő hirdetési szöveggel együtt az Alaplap szerkesztőségéhez küldjék el: 1251 Budapest, Pf. 71.

A PC Turbo Klub tagjai ebben a rovatban 20%-os kedvezménnyel hirdethetnek!

ADOK

Enterprise programok eladók. Válaszborítók ellenében listát küldök. 2000 program, sok kedvezmény, ajándék. Cím: Zemen László, 1164 Budapest XVI., Áföldi Ltp. 1035. Ép. II. Lh. TT. 7.

Enterprise-osok figyelem! Közel 2000 program megbízható minőségben, olcsón eladó 3,5"-os és 5,25"-os lemezekre is. Széles programválaszték a régebbi és a legújabb játék, felhasználói és egyéb programok között. Programcsere esetén listát, programvásárlásánál válaszborítékot kérek. Cím: Tóth Gusztáv, 1156 Bp. XV., Nádastópark 32.

Eladó **Commodore-64,** floppyvezérlővel, lemezekkel, programokkal. Tel.: Heinbach József, (06-27) 42-600.

C-64-re színvonalas játék- és felhasználói programok kaphatók lemezen, az áruk 5-15 Ft. A programokról részletes leírást küldünk. Válaszboríték nem szükséges. Cím: Lőrincz Endre, 6800 Hódmezővásárhely Somogyi u. 20.

Amiga 500-as 1 MB RAM-mal 40 000 forintért eladó. Ugyanitt eladó 4 MB-os RAM bővítő, 1,5 MB-ig kiépítve 22 000 forintért eladó. Cím: Major Nándor, Csorna, Kossuth u. 58. Tel.: Csorna 259. (munkahely)

CLIPPER 87 EXE és OVL programok visszaalakítása forrás formátumúvá saját fejlesztésű programmal, 100%-os hibamentességgel. Cím: DECOMPILER STUDIO, 6001 Kecskemét, Pf.: 298. Tel.: (06-76) 22-888, telefax: (76) 21-181, telex: (76) 26-515.

100-nál több **IBM játékprogram** olcsón másolható, esetleg cserélhető. Érdeklődni levélben vagy telefonon lehet. Cím: Dudás Péter, 1225 Bp. XXII., Csut u. 8/a. Tel.: 226-3376

Sürgősen eladó IBM PC-hez használható **lemezes alrendszer**: 1.2 MB floppy, 2x20 MB winchestercsatoló. Irányár: 25 000 Ft. Cím: Kráner Ferenc, 7570 Barcs, Darányi u. 6. Tel.: (06-83) 12-819.

CSERÉLEK

TOTÓ/LOTTÓ programcsere PC-re, C-64-re. Válaszborítékot kérek! Cím: Nagy József, 4031 Debrecen Derék u. 36. VII. 23.

Norton, Peter — Socha, John:

Az IBM PC Assembly nyelvű programozása

(Budapest, 1991. Novotrade Kiadó — Prentice Hall, 466 oldal. Ára: 698,– Ft.)

A szerzők azt ígérik, hogy mire az olvasó „befejezi e könyvet, képes lesz olyan teljes Assembly nyelvű programok elkészítésére, mint például szövegszerkesztők, segédprogramok és hasonlók". S mindehhez az indulásnál csupán annyit feltételeznek, hogy az olvasónak „van némi Basic-alapismerete, és képes dolgozni saját számítógépén".

Valóban, teljesen az alapoknál kezdik: a számrendszerek, az adatábrázolás, a gépi nyelv ismertetésével, s lépésről lépésre vezetnek el olyan „lemezfoldozó" program elkészítéséhez, mint a *Dskpatch*. Ennek — a végső lefordított formájában kb. 1 kbájt méretű — programnak a megvalósításán keresztül vezetik be a kezdetben járatlan olvasót a profi programozási módszerekbe. A készülő program minden fázisáról találunk jól áttekinthető forráslistát.

A könyv négy részre oszlik, e részek mindegyikében más a fő téma. Az első rész az alapokkal foglalkozik, megismertet a DEBUG használatával, ezen keresztül a gépi kódú programozás elemeivel. A második részben bemutatja az Assembly nyelvű programozás alapjait és az *assemblerek* használatát. Itt mutatják be a lépésenkénti fejlesztés, a moduláris programozás technikáját is. A harmadik rész fő témája hibakeresési és -javítási technika. A befejező rész már profi szintű módszerekkel foglalkozik, egyebek közt BIOS-hívásokkal is.



A függelékben megtaláljuk a könyvhöz *nem* mellékelt lemez részletes ismertetését, a lemezen található programok forráslistáját, a rendszerprogramok fontosabb hibaüzeneteit és a fontosabb táblázatokat: ASCII karakterkódokat, színkódokat, kiterjesztett billentyűkódokat, továbbá a címzési módokat és a fontosabb DOS-hívások funkciókódjait.

Nincsenek a könyvben assembler utasítástáblázatok, ezek a felhasználói kézikönyvben megtalálhatók. Nagyon hiányzik a makrók, illetve használatuk ismertetése. Sikerült néhány zavaró sajtóhibára is rábukkannom, csak remélni merem, hogy súlyos következményekkel járó hiba nincs a könyvben. A 153. oldal alján lévő „DEBUG lista" utolsó sora arra enged következtetni, hogy — a Novotrade Kiadó korábbi kiadványaiból „megszokott" módon — manipulált „gépi" listákkal van dolgunk.

(bl)

Bibliográfia

Összeállításunkban — a hónap témájához kapcsolódva — számítógépes és „számítógépesíthető" játékokról szóló könyvek közül válogattunk.

1001 Amiga. Játékleírások, Amiga és Amiga DOS, Felhasználói programok.
Budapest, 1990. LSI Oktatóközpont, 154 oldal. Ára: 253,– Ft.
1001/1 játék. C64/128 + Graphic Basic.
Budapest, 1988. LSI ATSZ, 115 oldal. Ára: 111,– Ft.
1001/2 játék. C64/128.
Budapest, 1989. LSI ATSZ, 134 oldal. Ára: 99,– Ft.
1001/3 játék. C64/128.
Budapest, 1989. LSI ATSZ, 161 oldal. Ára: 155,– Ft.
1001/4 játék. C64/128.
Budapest, 1989. LSI ATSZ, 157 oldal. Ára: 168,– Ft.
1001/5 játék. C64/128 + Amiga.
Budapest, 1988. LSI ATSZ, 219 oldal. Ára: 256,– Ft.
100+4 — Játékok és felhasználói programok. C16 — Plus/4.
Budapest, 1987. LSI ATSZ. 184 oldal. Ára: 161,– Ft.
100+4/2 — Játékok és felhasználói programok. C16 — Plus/4.
Budapest, 1988. LSI ATSZ, 209 oldal. Ára: 169,– Ft
100+4/3 — Játékok és felhasználói programok. C16 — Plus/4.
Budapest, 1989. LSI ATSZ, 111 oldal. Ára: 137,– Ft.
A játék matematikája. (Általános iskolai szakköri füzet)
Budapest, 1978. Tankönyvkiadó, 136 oldal. Ára: 8,– Ft.
Bartel — Kraas — Schrüfer: Számítógép és sakk.
Budapest, 1987. Data Becker — Novotrade, 333 oldal. Ára: 390,– Ft.
Csákány Antal — Dr. Vajda Ferenc: Játékok számítógéppel.
Budapest, 1985. Műszaki Könyvkiadó, 284 oldal. Ára: 57,– Ft.
Csupa játék ZX-Spectrumra.
Budapest, 1985. Műszaki Könyvkiadó, 77 oldal. Ára: 28,– Ft.
DaCosta, F.: A kalandprogram írásának rejtelmei. Hogyan írjunk BASIC nyelven az iskolaszámítógépre kalandprogramot.
Budapest, 1986. Műszaki Könyvkiadó, 232 oldal. Ára: 63,– Ft.
Dienes professzor játékai.
Budapest, 1989. Műszaki Könyvkiadó, 160 oldal + melléklet. Ára: 98,– Ft.
Eigen, Manfred — Winkler, Ruthild: A játék. Természeti törvények irányítják a véletlent.
Budapest, 1981. Gondolat, 426 oldal. Ára: 79,– Ft.
Etűdök személyi számítógépekre.
Budapest, 1984. Gondolat, 166 oldal. Ára: 67,– Ft.
Fábián István — Kiss László: 1001/5 játék. C64/128 — Amiga.
Budapest, 1989. LSI ATSZ, 220 oldal. Ára: 256,– Ft.
Korgyemszkij, B. A.: Matematikai fejtörők.
Budapest, 1962. Gondolat, 597 oldal. Ára: 37,– Ft.
Könözsi László — Gál István: Csupa szuperjáték C-PLUS/4, C-16, C-116 számítógépekre.
Budapest 1989. Műszaki Könyvkiadó, 171 oldal. Ára: 135,– Ft.
Lukács Ernőné — Tarján Rezsőné: Játékos matematika.
Budapest, 1975. Gondolat. 309 oldal. Ára: 20,– Ft.
Matematikai játékok. (Általános iskolai szakköri füzet)
Budapest, 1971. Tankönyvkiadó. 140 oldal. Ára: 8,– Ft.
Nagy játékkönyv Commodore 64-re. 28 szuperjáték és három meglepetés.
Budapest, 1990. Computer Panoráma, 102 oldal. Ára: 460,– + 90,– Ft.
Paszkalev, Zs.: Elektronikus logikai játékok.
Budapest, 1984. Műszaki Könyvkiadó, 113 oldal. Ára: 30,– Ft.
Sinclair Spectrum játék és program I.
Budapest, 1986. LSI ATSZ, 280 oldal. Ára: 194,– Ft.
Sinclair Spectrum játék és program II.
Budapest, 1986. LSI ATSZ, 112 oldal. Ára: 98,– Ft.
Sinclair Spectrum játék és program III.
Budapest, 1987. LSI ATSZ, 174 oldal. Ára: 120,– Ft.
Sinclair Spectrum játék és program 4.
Budapest, 1988. LSI ATSZ, 142 oldal + melléklet. Ára: 157,– Ft.
Sinclair Spectrum játék és program 5.
Budapest, 1988. LSI ATSZ, 179 oldal. Ára: 189,– Ft.
Sinclair Spectrum játék és program 6.
Budapest, 1989. LSI ATSZ, 189 oldal. Ára: 199,– Ft.
Spencer, Donald D.: Játékok BASIC nyelven.
Budapest, 1983. Számítástechnika-alkalmazási Vállalat. 191 oldal. Ára: 78,– Ft.
Vargha Balázs: Játékkoktél.
Budapest, 1967. Minerva, 291 oldal. Ára: 31,– Ft.

Know-how a Time Manager Internationaltól

Menedzseld magad!

Két régi gondolattömörítvénynek, a delphoi jósda Apolló templomában olvasható felszólításnak — „Ismerd meg önmagad!" — és az újabb keletű, de szintén klasszikus — „Az idő pénz." — tőmondatnak az ötvözése adja az alapot a TMI cég termékei kidolgozásának filozófiájához. Erre épít a „tmi key results" nevű szoftver is.

Egy eredményközpontú eszközhöz két elem szükséges: döntési háttér és az időbeosztáshoz naptár — valamint kell az is, hogy összefüggést teremtsünk a kettő között. Azért, hogy a célok határozzák meg az időbeosztást, ne pedig azok a követelmények, amelyeket a környezet támaszt.

A helyes célok kitűzéséhez elengedhetelen az önismeret, az elérésükhöz pedig erőforrások kellenek. Az idő — akár tőkésítve is — az egyik legfontosabb dolog, aminek tengerén „hajózni muszáj", s nem is akárhogy navigálva. Ezt tükrözi a TM-tanfolyam kézikönyvének tagolt címe:

Én
Az én időm
Az én életem

A Time Manager mint a személyes tervezést segítő eszköz három részből áll:

1) naptár rész;
2) döntési háttér rész;
3) vegyes kiegészítők.

A jel- és színrendszer az egyébként is logikus és áttekinthető kezelést tovább könnyíti; a szoftver menüinek és a funkciógombok használatának rutinját fokozza. (Az elhelyezés, az írásra rendelkezésre álló felületek aránya még azoknak is fogódzó a gyakorlatban, akik esetleg csak fekete-fehérben tudnak egyelőre dolgozni a monitorukon.)

A Time Manager módszereket, ötleteket és praktikus segédleteket ad ahhoz, hogy tulajdonosa eredményes legyen, mindig és mindenkor áttekintse és irányítsa az életét, emlékezzen mindenre, amire kell, javuljanak a kapcsolatai a hivatalos és a magánéletében, fokozódjék a kreativitása, összhangba kerüljenek törekvései és teljesítményei, rugalmasabban és lezserebben alkalmazkodják az őt érintő változásokhoz.

Már az maga is figyelmesség az ésszerűség mellett, hogy a kurzusokat szerdán és csütörtökön tartják: ellopni napokat egy tanfolyamra (agyonprogramozott vezetőktől, ügyvezetőktől, főmunkatársaktól?) — ezt csak a hét közepén szabad; el tudja indítani, és majd módja legyen be is fejezni/ellenőrizni a dolgait az embernek...

A kétnapos tanfolyam, mely belekóstoltatja a hallgatókat az élet ez új minőségének élvezetébe, valamint megtanítja a TM-eszközök variáns használatát, tulajdonképpen egy nagyszerű dán előadónak, Steen Sneidernek a showja. Felszabadult, látszólag kötetlen előadásmódja — akár egy kitűnő színész önálló produkcióján a közönséget: — magával ragad mindenkit, s ezáltal el is éri a bevésődést. Az egész „műsor" vidám élményként adja meg a célszerű ön-menedzsment elkezdéséhez az attitűdöt.

Érdekes kérdés, hogy a Time Manager International mint 16 éves múltú, a vezetés- és személyiségfejlesztő technikák kidolgozásával és ezek terjesztésével foglalkozó vállalat hogyan került kapcsolatba a Számalkkal? Vajon miért egy kifejezetten számítástechnikai profilú intézménnyel dolgoznak közösen?

A TMI már korábban is benn volt a magyar piacon, különböző vállalatoknak tartottak tanfolyamokat, sőt: mostanság is folynak vagy előkészületben vannak ilyenek (például a Malévnél). A Számalk viszont a nyílt szemináriumok megrendezésére vállalkozott, tehát ezekre egyenként, akár magánkezdeményezésre is jelentkezhet bárki, vagy olyan kis csoport egy-egy cégtől, akikért nem lenne kifizetődő helyi kurzust tartani. Egyébként nem szokványos ez az út, mert általában leányvállalataik vagy saját ügynökségeik vannak a „meghódított" országokban. Itt, Magyarországon a kooperációs lehetőséget hatékonyabbnak remélték, biztosítékul a Számalknál felhalmozott oktatási tapasztalat szolgált.

A Time Manager eredetileg hagyományos kivitelű segédeszköz: fizikailag papír és műanyag kombinációja (az információhordozókat és a „szerszámokat" tekintve is). Úgyszintén megszokott az aktatáska is, amely szinte státusszimbólum az elfoglalt, fontos emberek körében. Rendkívül imponáló ebből elővenni a T. M.-t, s forgatni azt... Ma már azonban az igazán korszerű eszközt a PC — s abból is a laptop — testesíti meg az üzleti világban.

A TMI az esetleges konkurenciát megelőzve 2-3 éve jelentkezett a számítógépi változattal, de azóta is finomítják, változtatják az igényeknek megfelelően — természetesen megtartva a korrelációt a hagyományos eszközök fejlesztése, valamint a program között. A „konzervatív" rendszert egyáltalán nem kívánják sutba dobni, hiszen a kettő remekül kiegészíti egymást — egyik lehet az irodában, másik otthon, egyik szolgálhat a mindennapokon, másik hivatalos utazáskor stb.

J. Á.

A T. M.-tanfolyamot (egyénileg vagy csoportnak) a Számalk Oktatási Irodáján lehet megrendelni — a hagyományos és/vagy a szoftveres változat megismerésére. Ugyanők a „tmi key results" program forgalmazói is, mely termék XT/AT-kompatibilis gépen fut, DOS 2.0 vagy e fölötti változatú operációs rendszer alatt. 512 kbájt központi memóriaszükséglettel kell számolni; fekete-fehér vagy színes monitor egyaránt megfelel; a szoftvernek a 3,5-es és az 5,25-ös floppys változata is megvásárolható. Menüs, funkciógombos, helpes rendszer, melyhez az angol nyelvű dokumentációval együtt 25 900 Ft-ért lehet hozzájutni — s mivel oktatási anyag, ÁFA nem növeli ezt a költséget.

viet 1991 INTERNATIONALE FACHMESSE FÜR ELEKTROTECHNIK UND INDUSTRIELLE ELEKTRONIK

Kong az ürességtől

Az ősz folyamán szép számmal rendeztek — itthon és külföldön — számítástechnikai kiállításokat. A nagy érdeklődést kiváltó budapesti UNIX kiállítás után nem sokkal a Budapest Kongresszusi Központ adott otthont a CAMP '91 nemzetközi kiállításnak, amelyet a tervek szerint ezentúl minden ősszel megrendeznek Magyarországon. Alig egy héttel később, a kétévenkénti bécsi VIET 1991 (Bécsi Ipari Elektronikai és Elektrotechnikai Kiállítás) keretében az immár hagyományos interCAD'91 kiállítás várta a CAD „szerelmeseit".

A kiállításon látottak azonban csalódást okoztak. Egy meglehetősen kis méretű pavilonban elsősorban osztrák kiállítók mutattak be CAD rendszereket, azokból is java részt az unalomig ismert képernyők köszöntek ránk.

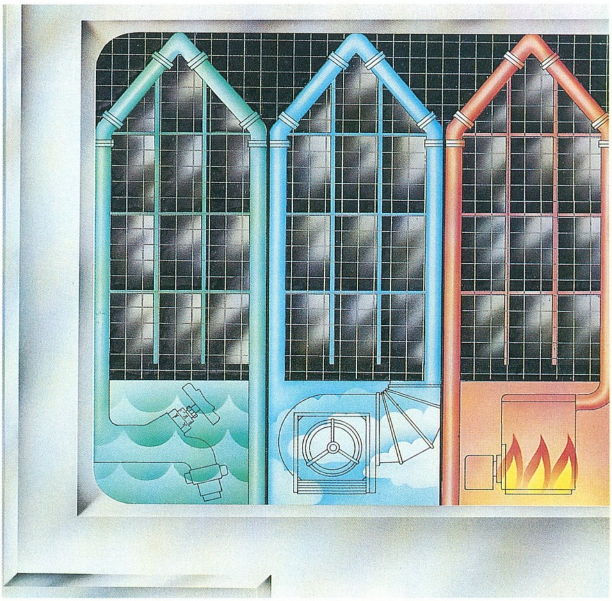
Így hiába mutatkozott be Bécsben is a CAD-világ csaknem teljes metszete, igazolódott, hogy kevesebb — mármint az őszi kiállításból — talán több lenne. Mind IBM PC-re, mind Macintoshra szép számmal láthattunk elektronikai, építészeti, gépészeti tervezőrendszereket — a megszokott „arcokat". Meglepő volt azonban az építészeti szoftverek túlsúlya. Ezek között örömmel fedeztük fel a Graphisoft által fejlesztett ArchiCAD és topCAD rendszereket (ha már régi ismerős, legalább magyar). A Magyarországon kevéssé elterjedt, ám külföldön nagy népszerűségnek örvendő ArchiCAD az osztrák építészeknek is kedvelt tervezési segédeszköze. A Graphisoftot képviselő osztrák A-null cég Apple gépeken dolgozó mérnökök munkáját segítő szoftverek egész sorát mutatta be. Így például a FOCUS 3D-s segédprogramot, amellyel egyszerűbb dolgok (nyílászárók, berendezési tárgyak) könnyedén tervezhetők és áthelyezhetők ArchiCAD-be.

Újdonságnak számított — azért ilyen is akadt elvétve — az olasz CUMTERRA építészeti szoftver, amellyel elsősorban a földrajzi környezetnek (hegyvidék, dombok, lankák) megfelelően tudjuk épületünket megtervezni. Kifejezetten geometriai CAD szoftvert (MapGraphix) is bemutattak Macintoshra. Hogy az építészeti szolgáltatások teljessé váljanak Mac-gépekre, belső építészeti szoftver sem hiányozhatott (ZOOM) a repertoárból. Compaq gépeken mutatták be az ismert MICADO 3D-s építészeti szoftvert, amely Németország, Ausztria és Svájc mellett már Lengyelországban és Csehszlovákiában is népszerű az építészek körében. A CAD-palettát színesítette a triCAD szoftver, amely elsősorban a csőtervezés hatékony segédeszköze.

A CAD világától elválaszthatatlanok a tervek megjelenítésére szolgáló plotterek. A különböző méretű (A/2, A/3) és eltérő tollszámú (4-8 stift) asztali plotterek mellett többféle álló (A/0, A/1) plotter is szerepelt. A Hewlett-Packard és Roland gyártmányú plottercsaládok mellett a Graphtec is szép számmal állított ki plottereket. Újdonságként — már a második! — szerepeltek a Graphtec vágó- és betűplotterei, amelyekbe akár egy 50 méter hosszú, speciális öntapadós papírt is befűzhetünk, és amelyből tetszőleges alakzatokat, szövegeket és többféle betűtípust is ki lehet vágni.

Summázva a látottakat: az interCAD '91 kiállítás látványos volt — milyen is lenne egy CAD kiállítás —, ám főleg jól ismert szoftverek szerepeltek, kevés volt az újdonság, és a kiállításlátogatók sem tolongtak a standokon. Úgy gondoljuk, ha nem ilyen sűrűn követnék egymást a hasonló témájú kiállítások, talán a szakma is nagyobb lelkesedéssel készülne egy-egy újabb bemutatóra.

Sziebig Andrea



A „Unix szelleme" járta be Budapestet

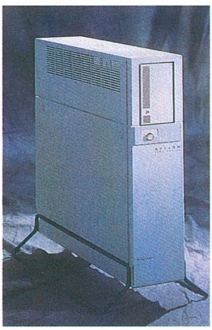
A nyári uborkaszezon után szeptembertől meglehetősen felpezsdült a hazai számítástechnikai élet.
Kiállítások, termékbemutatók követték egymást.
Ezek közül az egyik legjelentősebb a nemzetközi Unix-kiállítás volt, ahol a világ élvonalába tartozó, Unix-alapú nyitott rendszerekkel ismerkedhettek meg a látogatók.
Az itt bemutatott hardverújdonságok egy részéről számolunk most be, emellett hírt adunk egy környezetbarát szolgáltatásról a hazai számítástechnikában.

Egy szervercsalád ringbe száll

A szeptember végi kiállításon a számos Unix-alapú szoftverrendszer mellett szép számmal mutattak be Magyarországon eddig még nem látott vagy csak kevéssé ismert gépcsaládot. Ilyen volt például a Data General AViiON ICL szervercsaládja, a SEQUENT számítógépcsalád DRS6000 Unix szervere vagy a Sun család két legkisebb tagja.

Az AViiON termékeiről megtudtuk, hogy jelenleg a RISC alapú Unix gépek között ez a gépcsalád képviseli a legjobb ár/teljesítmény viszonyt. Ezek, a Microsystem által forgalmazott szerverek ideálisan alkalmazhatók elsősorban pénzintézeti, vállalati, államigazgatási, valamint egészségügyi rendszerek kiszolgáló állomásaként.

Az AViiON hostok lehetővé teszik, hogy a felhasználó tetszőlegesen más géptípussal is kommunikáljon. Így a már meglévő, lokális feldolgozásra alkalmas berendezések egységes hálózatba integrálhatók. Ezáltal új távlatok nyílnak meg egy egységes információs rendszer felé, amelyhez különféle hálózatkezelő eszközök (Ethernet, Novell, WAN,...) állnak rendelkezésre. Mivel az AViiON berendezések beépíthetők a hazai Novell-hálózatokba, így a sok helyen felmerülő „hogyan tovább" kérdésre is nyújt megoldást ez a szervercsalád is.



Mivel a kínálat tagjai széles teljesítményskálán helyezkednek el, lehetővé válik a megfelelő kapacitású rendszer kiválasztása. A különböző géptípusok között teljes a szoftverkompatibilitás. Egyszerű kártyacserével a felhasználó az egyprocesszorosról kettő- vagy négyprocesszorosra bővítheti rendszerét. A felhasználó a teljesítménynövelés mellett továbbra is használhatja a régebbi perifériáit és a gépen futó alkalmazásokat.

A családot alkotó gépek cash-memóriavezérlőt is tartalmaznak; központi egységük a Motorola 88100 RISC alapú mikroprocesszora, amely 32 bites regiszterekkel, adatutakkal és címekkel dolgozik. A szabványok fontos szerepet játszanak: például szabványosított SCSI interfészű háttértárak használatosak ezekben a hostokban. Az AViiON gépeken futó DG/UX operációs rendszer — amely vezető szerepet tölt be a Unix-implementációk körében — biztosítja a szoftver hordozhatóságát. Az ipari szabványoknak eleget tevő alkalmazói rendszerek széles köre futtatható a gépeken. Több mint 400 gyártó 1500 szoftverterméke alkalmazható a család gépein.

A kiállításon a Microsystem szakemberei elmondták, hogy elsősorban a közepes méretű vállalatokat célozzák meg a hazai piacon. Ugyanis egy ilyen nagyságú vállalat teljes információs rendszerének kiszolgálása egy közel 3 millió forintos konfigurációval (33 MHz, 39 MIPS, 32 MB RAM, 1 GB hard diszk, 525 MB szalag, Ethernet-csatoló) már megvalósítható.

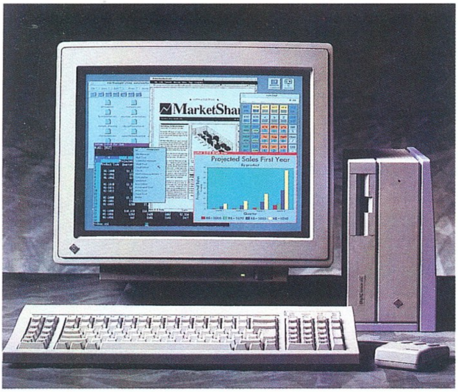
Kíváncsian várjuk, hogy a kitűnő paraméterekkel rendelkező, kedvező árfekvésű szervercsalád milyen sikereket ér el a nagy megbízhatóságot elváró, igényes felhasználók körében.

„Ragyogó" workstation-piac

Ha már a nyitott rendszerekről esett szó, akkor feltétlenül meg kell említenünk a Sun céget, amely jelentős szerepet játszik az Open Systems világban. Az általuk kifejlesztett SPARC processzorok nemzetközi szabványok lettek, ugyanakkor a Sun más nemzetközi szabványokat is beépít gépeibe. Így tágabb piaci lehetőségek nyíltak meg a Sun gépek előtt. Ennek a nyitottságnak köszönhető, hogy az úgynevezett SPARCware környezetben futó alkalmazói szoftverek száma ma már 3200 fölött van.

A Sun gépek hivatalos magyarországi viszonteladójától (ez az MTA SZTAKI Elektronikai Fejlesztési Osztálya) megtudtuk, hogy Unix-környezetbe szeretnék integrálni mindazokat a grafikushardver- és -szoftverfejlesztéseket, amelyekkel hosszú évek óta foglalkoznak.

A Unix-kiállításon a Sun választékából csak a két legkisebb elemet ismerhették meg az érdeklődők. Ezek egyike a desktop-kivitelű SPARCstation SLC, amely alapkiépítésben diszk nélküli, monokróm workstation. A cég desktop-spekt-



rumának másik fénye a SPARCstation IPC — ez mind monokróm, mind színes változatban hozzáférhető. Az egyik ilyen gépre, a SPARCstation IPC/LPC munkaállomásra az év végéig kedvezményes vásárlási akciót is hirdettek a SZTAKI-sok. Ez az egyetlen olyan Sun gép, amely nincs COCOM-engedélyhez kötve, így 1 hónapos határidővel ígérik a szállítást. A 8 MB memóriát, 207 MB harddiszket tartalmazó konfigurációt 16"-os színes monitorral, 1,44 MB floppyval, Sun CD-ROM egységgel, az operációs rendszer dokumentációjával együtt közel 900 000 forintért kínálják az érdeklődőknek.

A jelenleg is kapható gépek mellett hamarosan a hazai piacon is hozzáférhető lesz a Sun munkaállomások csúcsmodellje, a SPARCstation 2. A kiállításon megtudtuk azt is, hogy a Sun gépek mindkét családja (Desk Top, Desk Side) megfelelő kiépítésben szerverként is használható; rendszerszoftverük a SunOS, amely a Unix-családba tartozó szoftver. Mivel a Sun a heterogén hálózatok felé is nyitott, így — ismerve a hazai hálózati viszonyokat — ez nagy segítséget jelenthet a Sun gépek magyarországi fogadtatásában.

Természetkímélés a számítástechnikában?

A munka (és a pazarlás) e torz termései, az irodai hulladékok műszakilag még felhasználható, újra hasznosítható anyagok. Ennek ellenére az ilyen, a „kényesebb" szobákban, termekben fölöslegessé vált limlomok eddig a szemétre kerültek.

A nehezen lebomló, nagyméretű, főleg műanyag holmik környezetszennyező hatása mindannyiunk számára nyilvánvaló. Azonban mégiscsak újabb (pénz)pazarlás szemétre dobni a még felhasználható anyagokat! Ennek a kettősségnek a feloldását kínálja a felhasználóknak a FARBAX Kft. a lézernyomtatókban, fénymásolókban használatos festékkazetták és fotohengerek felújítását kínáló szolgáltatásával.

Vegyük például az egyik legelterjedtebb lézenyomtatót, a HP LaserJet III típusú lézerprinter EP-S kazettáját! A kazetta újonnan 10 500 forintba kerül (+25% ÁFA), felújítva 6 877 Ft (+15% ÁFA). Sőt, ha a vevő még üres kazettát is ad, akkor további 300 forinttal csökken a számla. A festékkazettákat cserélhetjünk akár személyesen is a szeptemberben megnyitott bemutatótermükben. De telefonhívásra Budapesten, sőt sok esetben vidéken is díjtalanul kiszállítják a „second hand" kazettákat.

Egyébként senki sem a saját felújított kazettáját kapja vissza — hacsak egy-szer-egyszer egészen véletlenül nem —, ugyanis a restaurálásra a szombathelyi Refill Service-ben találtak lehetőséget. A felújítást svájci gépsoron, japán festékporok felhasználásával, a svájci FARBAX-technológia alapján végzik el. A kazettákat darabokra szedik szét, az alkatrészeket egyenként megvizsgálják, a kopottat, karcosat tökéletessel pótolják. A fotohengert is megvizsgálják, és szükség esetén ennek a helyére is kifogástalan kerül.

Az összeszerelés, festékporral való feltöltés után valamennyi kazettát tesztelik. Végül légmentesen, fémfóliával lehegesztik a kazettákat. Így az eredetivel azonos minőségű kazettát kap az ügyfél, amelyre 1 év garanciát is ad a FARBAX Kft. Mivel egy kazetta a kopó alkatrészek cseréjével elvileg végtelen sokszor felújítható, tehát e téren a természetrombolás — legalább egy ideig — így fékezhetővé válik.

Sziebig Andrea




NTT—2000
Trade and Service, Ltd.
A RANK XEROX
hivatalos dealere
RANK XEROX
fénymásolók, faxok,
lézerprinterek,
írógépek forgalmazása
RANK XEROX
berendezések
színvonalas szervizellátása
RANK XEROX
kellékanyagok biztosítása
Cím: 1431 Budapest
VIII., Mária u. 20.
Telefon: 1340-900, 1340-393
Fax: 1340-568
XEROX 7009
INFORMÁCIÓKÉRÉS: 08 ▲

Ha a megbízhatóság a döntő...



VIGYÁZAT! Jól bevezetett és hírnévnek örvendő márkanevünkkel
kétes minőségű, hasonló hangzású nevek élnek vissza!

Forgalmazó:

INTERAG

INTERAG INFORMATIKA
Budapest 1136 Pannónia u. 11.
Tel./fax.: 132-9375 Molnár Péter, Sugár Mihály

People Committed To InfoTech

MITAC

INFORMÁCIÓKÉRÉS: 0 ▶


5. Presentation-quality graphics
Annual Sales
A wide variety of colors and fonts add that professional touch.
SONICA
ALR
Köszönjük, hogy a COMPFAIR 91 kiállításon megtiszteltek látogatásukkal!
NETCOM COMPUTER
1061. BP. PAULAY 22-24. TEL/FAX: 1427580